ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

DESARROLLO DE UNA APLICACIÓN TIPO RED SOCIAL DE "ESTILOS DE ROPA"

DESARROLLO DEL BACKEND

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE

ERICK ESTIVEN RUIZ CHILUISA

erick.ruiz@epn.edu.ec

DIRECTOR: IVONNE FERNANDA MALDONADO SOLIZ

ivonne.maldonadof@epn.edu.ec

Quito, septiembre 2024

CERTIFICACIONES

Yo, Erick Estiven Ruiz Chiluisa declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Erick Estiven Ruiz Chiluisa

Erick.ruiz@epn.edu.ec

Ruizerick2525@gmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por Erick Estiven Ruiz Chiluisa, bajo mi supervisión.

Ivonne Fernanda Maldonado Soliz

DIRECTOR

ivonne.maldonadof@epn.edu.ec

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Erick Estiven Ruiz Chiluisa

DEDICATORIA

Este trabajo lo dedico a las personas que han creído en mí y me han apoyado durante todo este tiempo, pero lo dedico en especial a mi abuela materna María Rodríguez en paz descansa, mujer que me educo y cuido durante mi infancia, y apoyo durante mi crecimiento, espero que esté orgullosa por lo que he conseguido hasta ahora y me sigua cuidando donde este.

A mis amigos más cercanos, Gilmar, Paul, Erika, David y Madelin, con los que he pasado mis mejores momentos, y me han permitido crecer como persona, son los mejores amigos que he tenido.

Erick Ruiz.

AGRADECIMIENTO

Agradezco a familia, por el apoyo que me han brindado en este proceso, gracias a ellos soy lo que soy ahora, en lo bueno y en lo malo, mi familia siempre ha estado allí apoyándome.

Agradezco a los pocos amigos de la universidad, mismos que me han ayudado a crecer y entender mejor mis falencias, y permitiendo mejorar en cada uno de los aspectos de mi vida, agradezco a Madelin mi amiga verdadera y mi fuente de inspiración.

A mis docentes de la ESFOT, gracias a ellos he aprendido una gran cantidad de conocimiento de los temas de mi interés, gracias por su dedicación y paciencia en este proceso, a mi directora de tesis Ivonne Maldonado, un rotundo gracias por su tiempo y paciencia en este último escalón de mi etapa universitaria.

Erick Ruiz.

ÍNDICE DE CONTENIDO

RESUMEN

Existen diversas aplicaciones relacionadas a la moda y estilos de vestir, mismas que presenta diversas funcionalidades, pero estas no presentan una gran interactividad entre los usuarios dado que se centran en la compra/venta de dichos estilos de vestir, por lo cual se propone el desarrollo de una Aplicación de Tipo Red Social de “Estilos de Ropa” con el objetivo de mejorar la interacción entre los usuarios aficionados a los estilos de vestir.

La implementación del backend ROPDAT presenta diversas funcionalidades que garantizan la interacción de la aplicación móvil, y la gestión de la aplicación por medio de una aplicación web. El componente backend incluye un autenticador por medio de JSON WEB Tokens reforzando la seguridad tanto en la parte móvil como en la parte web, además de usar Node.js y Render para asegurar la eficacia y eficiencia del componente en su desarrollo y despliegue. El componente ha sido sometido a pruebas de Funcionalidad, Carga y Rendimiento para garantizar su correcto funcionamiento en relación a los requerimientos establecidos, por lo que se centra en ofrecer un API Rest para el consumo del componente web y móvil con la finalidad de ofrecer una aplicación segura, interactiva y atractiva para los usuarios.

El presenta documento se organiza por secciones, primera: descripción del componente, objetivo general y específicos y marco teórico. Segunda: metodología de trabajo SCRUM, arquitectura y herramientas para el desarrollo del componente backend. Tercera: resultados de los Sprints desarrollados. Las secciones Cuarta y Quinta: presentan conclusiones y recomendaciones.

PALABRAS CLAVE: Backend, token, seguridad, Node.js, componente, autenticación, Red social.

ABSTRATCT

There are various applications related to fashion and clothing styles, which present various functionalities, but these do not present great interactivity between users since they focus on the purchase/sale of said clothing styles, which is why the development is proposed. of a Social Network Type Application of “Clothing Styles” with the aim of improving interaction between users who are fond of clothing styles.

The implementation of the ROPDAT backend presents various functionalities that guarantee the interaction, security of the mobile application, and the management of the application through a web application. The backend component includes an authenticator through JSON WEB Tokens, reinforcing security in both the mobile part and the web part, in addition to using Node.js and Render to ensure the effectiveness and efficiency of the component in its development and deployment. The component has been subjected to Functionality, Load and Performance tests to guarantee its correct operation in relation to the established requirements, so it focuses on offering a Rest API for the consumption of the web and mobile component with the purpose of offering an application safe, interactive and attractive for users.

This document is organized by sections, first: description of the component, general and specific objective and theoretical framework. Second: SCRUM work methodology, architecture and tools for the development of the backend component. Third: results of the developed Sprints. The Fourth and Fifth sections: present conclusions and recommendations.

KEYWORDS: Backend, token, security, Node.js, component, authentication, social networks.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

En la actualidad las tendencias y modas son visualizadas en diversos medios, siendo los principales las redes sociales, aunque también existen tipos de aplicaciones dedicadas específicamente a este mercado [1].

Al analizar aplicaciones como Bantoa, Your Closet, Combyne o 21 Buttons en las que se oferta outfits se encuentra una gran cantidad de actividades relacionadas a la compra/venta de productos, modas actuales y estilos más populares en la actualidad [1], sin embargo estas aplicaciones no ayudan con la demanda de asistentes de moda que cuenten con un ambiente de interacción entre usuarios aficionados a la moda, mismos que puedan compartir y visualizar de forma directa y organizada diversos estilos de ropa (clasificados por épocas, tendencias, temporadas, etc.,.); naciendo así la idea del desarrollo de una aplicación tipo red social de “estilos de ropa”.

El presente Trabajo de Integración Curricular detalla el desarrollo del componente backend para la aplicación antes mencionada, teniendo como objetivo principal el gestionar la información por medio de un API Rest tanto para el componente móvil como del componente web. La API Rest permite crear, editar, eliminar y visualizar publicaciones, clasificándolas de acuerdo con los criterios (Estilo del vestuario, Época del vestuario, Temporada del vestuario y el Género al que va dirigido el vestuario) que el mismo usuario selecciona, así como la interacción (me gusta, me disgusta, reportar) entre publicaciones de diferentes usuarios, acciones permitidas para los usuarios registrados. Por otro lado, también se cuenta con acciones que permiten un control, monitoreo y regulación de las diferentes publicaciones (por medio del usuario moderador) con el objetivo de evitar el ingreso de contenido inadecuado y bloquear cuentas que propaguen dicho contenido, fomentando así un ambiente adecuado para todos los usuarios.

En resumen, el componente backend ofrece las funcionalidades y es el soporte para que la aplicación tipo red social de “estilos de ropa” funcione de manera correcta tanto en el lado web como en el lado móvil, gestionando la entrada y salida de información, contando con tres roles: moderador e invitado (para la parte web) y usuario pertenece a la parte móvil.

1.1 Objetivo General

Desarrollar un backend para una aplicación tipo red social de “Estilos de Ropa”.

1.2 Objetivos Específicos

1. Determinar los requerimientos para el desarrollo del backend de la aplicación tipo red social de “Estilos de Ropa”.
2. Diseñar la colección y los modelos de para la base de datos no Relacional de la aplicación tipo red social de “Estilo de Ropa”.
3. Codificar los endpoints según los requerimientos.
4. Ejecutar las pruebas pertinentes al backend para verificar su funcionalidad.
5. Desplegar el backend hacia producción.

1.3 Alcance

Las redes sociales son utilizadas por gran parte de la población, cada una de ellas para objetivos en específico, pero a su vez permitiendo la interacción entre diferentes personas, satisfaciendo ciertas necesidades de los usuarios [2].

El componente backend bautizado ROPDAT presenta diversas funcionalidades distribuidas en roles (usuario, moderador e invitado) que de acuerdo con los diferentes permisos tienen la posibilidad de crear publicaciones de estilos de ropa, comentar publicaciones de otros usuarios, reaccionar a las diferentes publicaciones y controlar (bloquear, desbloquear, restringir o eliminar) cuentas y publicaciones. Además, ROPDAT presenta las funcionalidades principales de una red social tales como: registrarse, iniciar sesión y actualizar perfil.

El desarrollo del componente ha seguido la metodología de trabajo SCRUM, almacena sus datos en base de datos no relaciona, una herramienta en la nube para almacenar las imágenes o fotos, y trabajando con una arquitectura MVC (modelo vista-controlador). Así, para lograr su consumo exitoso, se ha desplegado mediante la herramienta RENDER y conectado a la base de datos en la nube MONGODB Atlas, junto con Cloudinary como la herramienta para almacenar las fotos o imágenes de los usuarios.

A continuación, se lista las funcionalidades por rol de usuario que se han desarrollado.

Roles de la aplicación:

- Usuario
- Moderador
- Invitado

El usuario puede:

- Iniciar y cerrar sesión
- Publicar sus estilos de ropa mediante fotos y descripción
- Comentar las publicaciones de otros usuarios
- Reaccionar a las publicaciones
- Visualizar las publicaciones globales

El moderador puede:

- Visualizar la lista de perfiles
- Bloquear cuentas que cometieron infracciones
- Eliminar cuentas de forma permanente

El invitado puede:

- Visualizar las publicaciones de la aplicación

1.4 Marco Teórico

Backend

En el desarrollo de aplicaciones se habla mucho de este componente, mismo que es considerado el servidor de las aplicaciones o sistemas, esto debido a que este componente procesa las solicitudes y devuelve una respuesta [3].

Este componente se encarga de dar funcionalidad y lógica a las acciones que posee una aplicación o sistema, tales como, conectividad y trabajo con una base de datos, envío de información por diversos medios, correo electrónico, mensajes, etc. [3].

El objetivo principal de un backend es proporcionar la información a mostrar dentro de una aplicación o frontend, además de procesar ciertas acciones dentro de la misma, pudiendo ser estas, cálculos matemáticos, envío de mensajes, envío de formularios, entre otros [3].

Base de datos no relacional

Este sistema de almacenamiento se organiza por medio de colecciones y documentos, permitiendo tener una gran escalabilidad y gestión de grandes volúmenes de datos [4].

La estructura de los documentos es flexible dentro de la base, permitiendo agregar diversos campos a cada documento, es por eso por lo que se denomina datos no estructurados o semiestructurados [4].

La forma de realizar consultas es por medio de diversas herramientas que faciliten la comunicación entre la base y el sistema, no trabaja con consultas SQL, pero puede ayudarse de ellas en forma de apoyo [4].

Cloudinary

Es un servicio en línea que permite almacenar y gestionar imágenes de un servidor y poder trabajar con ellas por medio de URLS personalizadas, ofrece diversas ventajas siendo la principal de ellas, reducir el tiempo de trabajo del servidor, optimizando las respuestas de este [5].

Otra ventaja es que permite la optimización de la calidad de las imágenes mediante la implementación de un parámetro, de tal forma tener la mejor imagen dependiendo de lo que necesitemos [5].

Node.js

Es un entorno de ejecución de JavaScript, mismo que permite ejecutar programas escritos con este lenguaje, otorgando diversos beneficios y solucionando diversos problemas, permitiendo un desarrollo más avanzado dentro de nuestros sistemas [6].

Trabaja por medio de un modelo de entrada y salida sin bloqueo, permitiendo el control de eventos siendo más ligero y eficiente, realizando diversas operaciones tales como, leer, escribir archivos o realizar solicitudes HTTP [6].

2 METODOLOGÍA

Un estudio de caso es un método de investigación cualitativo utilizado para investigar diversos fenómenos dentro de diferentes campos de la ciencia, caracterizado por la búsqueda exhaustiva de información sin centrarse en el estudio estadístico de dichos fenómenos, con la intención de elaborar hipótesis y teorías con la finalidad de realizar más investigaciones aún más complejas y elaboradas [7].

Esta metodología permite describir y registrar hechos o circunstancias de caso con la finalidad de comprobarlos y compararlos con situaciones o casos similares o relacionados, con la particularidad de que se centra en temas concretos y únicos, permitiendo el desarrollo de una investigación más eficaz [7].

El presente Trabajo de Integración Curricular adopta un enfoque de estudio de caso para explorar la implementación de un backend para una aplicación tipo red social de “Estilos de Ropa”. La investigación tiene por finalidad analizar la importancia y utilidad de las aplicaciones a desarrollar en base al análisis de diversas aplicaciones del mismo estilo, de tal forma que se asegura el desarrollo de un backend que sea funcional y eficiente para su consumo por parte de los componentes web y móvil.

2.1 Metodología de Desarrollo

La metodología de desarrollo para el presente Trabajo de Integración Curricular es SCRUM, una metodología ágil que permite gestionar proyectos de forma más eficiente, principalmente proyectos desarrollados en plazos de tiempo cortos, esta metodología ayuda a los equipos de trabajo a organizar de mejor manera la estructura y la gestión del proyecto mediante la combinación de principios y prácticas, además de utilizar diversas actividades, herramientas y funciones [8].

Trabajar con esta metodología requiere de diversas partes bien definidas, tales como, el equipo y los artefactos, siendo importantes para el correcto desarrollo del proyecto. La fortaleza esta metodología es la manera en que se distribuye el trabajo, siendo llamados sprints, cada uno con un objetivo en específico y un tiempo de trabajo establecido, de tal manera que se organice el tiempo de desarrollo de forma más eficaz para todo el equipo [8].

Roles

Dentro del desarrollo de la metodología SCRUM es necesario designar diversos roles importantes, mismos que garantizaran la eficiencia del proyecto, los roles principales de SCRUM para este proyecto son:

Product owner

Este rol asume la responsabilidad de maximizar la eficiencia del trabajo, además de gestionar las diversas actividades a realizar dentro del product backlog, manteniendo una constante comunicación con el cliente, conociendo la forma de trabajo y conocimiento sobre el negocio o empresa. Por ello tiene la autoridad sobre lo que se realiza y lo que debe ser entregado de cada uno de los sprints [9]. La persona designada para este rol se puede visualizar en **Tabla 2.1**.

Scrum Master

Este rol es el responsable del entendimiento de la metodología dentro del equipo de trabajo o desarrollo, de tal forma su rol se centra en organizar y explicar el desarrollo de los sprints, apoyando a los desarrolladores en el entendimiento de los sprints, mitigando inconvenientes o impedimentos [9]. El encargado de desempeñar este rol se visualiza en **Tabla 2.1**.

Development Team

Este rol se encarga de realizar y cumplir las tareas designadas por el Product Owner en el orden de prioridad designado, de tal forma este equipo se autoorganiza para desarrollar las actividades descritas en cada sprint, el mismo es único pues no cuenta con sub-equipos o alternos que colaboren con ellos, trasmitiendo la responsabilidad compartida a cada miembro en caso de existir algún fallo en el desarrollo [8]. El miembro del equipo de desarrollo designado se visualizará en **Tabla 2.1**.

Tabla 2.1: Roles designados

Rol	Integrantes
Product Owner	Ing. Ivonne Maldonado
Scrum Master	Ing. Ivonne Maldonado
Development Team	Ruiz Erick

Artefactos

Los artefactos SCRUM ofrecen la información necesaria para el desarrollo del proyecto por parte del equipo SCRUM, en estos artefactos se detalla las actividades a realizar y el tiempo en el que deben ser realizadas [8].

Recopilación de Requerimientos

Considerado uno de los procesos esenciales dentro del desarrollo de software, recopilar y analizar la información permite recopilar los requerimientos del usuario o usuarios dentro de un sistema o una aplicación, garantizando así que el desarrollo cumpla con todas las necesidades y expectativas del cliente [10].

En este proceso de recopilar los requerimientos, se debe identificar los objetivos del sistema o aplicación, sus requisitos funcionales y no funcionales, al fin de organizar de una forma eficiente el desarrollo a trabajar [10].

Este proceso dentro de SCRUM permite al equipo organizar de forma correcta los Sprints a realizarse para poder culminar con él y en el tiempo solicitado [10].

El levantamiento de requerimientos se detalla en el **Levantamiento de requerimientos**.

Historias de Usuario

Las historias de usuario se realizan con la finalidad de describir las características del sistema o aplicación desde la perspectiva del usuario, permitiendo detallar y explicar las funcionalidades de cada componente, permitiendo agregar observaciones, características únicas, etc. [11].

La **Tabla 2.2** presenta un ejemplo del formato utilizado para detallar las historias de usuarios. En el **Historias de usuario** se pueden visualizar todas las historias de usuarios

Tabla 2.2: Formato de Historias de Usuarios

HISTORIA DE USUARIO	
Identificador: HU001	**Usuario:** usuario, moderador
Nombre historia: Registro e inicio de sesión	
Prioridad en Aplicación: Alta	**Riesgo en Aplicación:** Alta
Iteración asignada: 1	
Responsable: Erick Ruiz	
Descripción: **Como:** Usuario y moderador **Quiero:** Registrarme e iniciar sesión **Para:** Hacer uso de la aplicación según su rol asignado	

Alcance:

1. El endpoint de registro para usuarios tendrá en el formulario los siguientes campos:
    a. Nombre
    b. Apellido
    c. Fecha de nacimiento
    d. Correo electrónico
    e. Contraseña
2. El endpoint de registro para moderadores tendrá en el formulario los siguientes campos:
    a. Nombre
    b. Apellido
    c. Correo electrónico
3. El endpoint de inicio de sesión para usuarios tendrá en el formulario los siguientes campos:
    a. Correo electrónico
    b. Contraseña
4. El endpoint de inicio de sesión para moderadores tendrá en el formulario los siguientes campos:
    a. Correo electrónico
    b. Contraseña
5. El endpoint Primer Inicio, solicita al moderador que en el primer inicio de sesión cambie la contraseña, ingresando un código único.

Observación:

- **Correo electrónico (campo no editable):** El correo ingresado por los usuarios y moderadores debe ser único independientemente de cada rol.
- **Nombre (Campo de texto):** Nombre o nombres del usuario.
- **Apellido (Campo de texto):** Apellido o apellidos del usuario.
- **Contraseña (Campo de texto):** Deberá cumplir con los criterios para crear una contraseña segura.
- **Fecha de nacimiento (Campo no editable):** Deberá cumplir con el requisito de mayoría de edad (+18) para usar la aplicación.
- **Código de moderador (Campo no editable):** Se genera de forma aleatoria y no será modificado, usado como única vez en el primer inicio de sesión del moderador.

Criterios de Aceptación:

- Todos los campos (Nombre, apellido, Fecha de nacimiento, email, contraseña y código de moderador) son obligatorios.
- Los usuarios recibirán un correo para poder verificar su cuenta e iniciar sesión dentro de la aplicación móvil.
- Los moderadores recibirán un correo con sus credenciales de ingreso (contraseña y código único) para ingresar la aplicación web.
- La creación de cuenta para cualquier rol solo será posible si se completan todos los campos.
- Los correos electrónicos son únicos para cada rol, un moderador no puede tener una cuenta de usuario con el mismo correo.

- El registro de moderadores se realizará únicamente por medio del moderador predefinido en la base de datos.
- Se enviarán mensajes error si por algún motivo no se puede crear la cuenta o validar el correo electrónico.
- Los moderadores deberán actualizar su contraseña en el primer ingreso, para lo cual utilizará el código único generado.
- Los moderadores podrán actualizar su contraseña de acuerdo a sus necesidades.
- Los endpoint de inicio de sesión serán únicos para cada aplicación (móvil y web)
- El rol moderador deberá cambiar su contraseña por primera vez cuando inicie sesión, esto para mantener un nivel de seguridad dentro de la aplicación web.

Product Backlog

El Product Backlog es considerado una lista con las características del componente a desarrollar, donde se detalla la prioridad del desarrollo de los requerimientos levantados, este listado se mantiene en actualización con la finalidad de señalar el progreso que se tiene en el desarrollo del componente, de tal forma que el Developer team pueda cumplir con las fechas establecidas [11]. En el **Product Backlog** se visualiza este artefacto.

Sprint Backlog

El Sprint Backlog es una subdivisión del Product Backlog que contiene Sprints requeridos para el desarrollo de la aplicación, cada Sprint contiene Historias de usuarios y un tiempo de desarrollo, de tal forma que permita a los desarrolladores organizar su tiempo de desarrollo para cumplir con las fechas establecidas y entregar Sprints de calidad [11]. En el **Sprint Backlog** se puede visualizar este artefacto.

2.2 Diseño de la arquitectura

La arquitectura brinda un apoyo a los desarrolladores, facilitando la planificación del desarrollo y la selección de herramientas a utilizar durante este proceso. El proceso de desarrollo de la arquitectura es crucial para definir los componentes a desarrollar, las decisiones que se toman en torno a este desarrollo, proporcionando un cimiento claro para el desarrollo del software [12].

Arquitectura de datos

Una arquitectura de datos se caracteriza en demostrar cómo es la gestión de los datos dentro de la aplicación, tanto el ingreso como la salida de dichos datos, y como estos son

consumidos o distribuidos, esta arquitectura de datos permite establecer como los datos se almacenan dentro del sistema de base de datos. La arquitectura de datos se restringe de acuerdo a los requerimientos de la aplicación o del negocio de manera que los modelos o tablas contienen la información requerida y esencial para su funcionalidad [13].

La **Figura 2.1** demuestra la estructura de datos que se ha implementado dentro de la base de datos no relacional, permitiendo visualizar todas las colecciones con las que trabaja el backend ROPDAT.

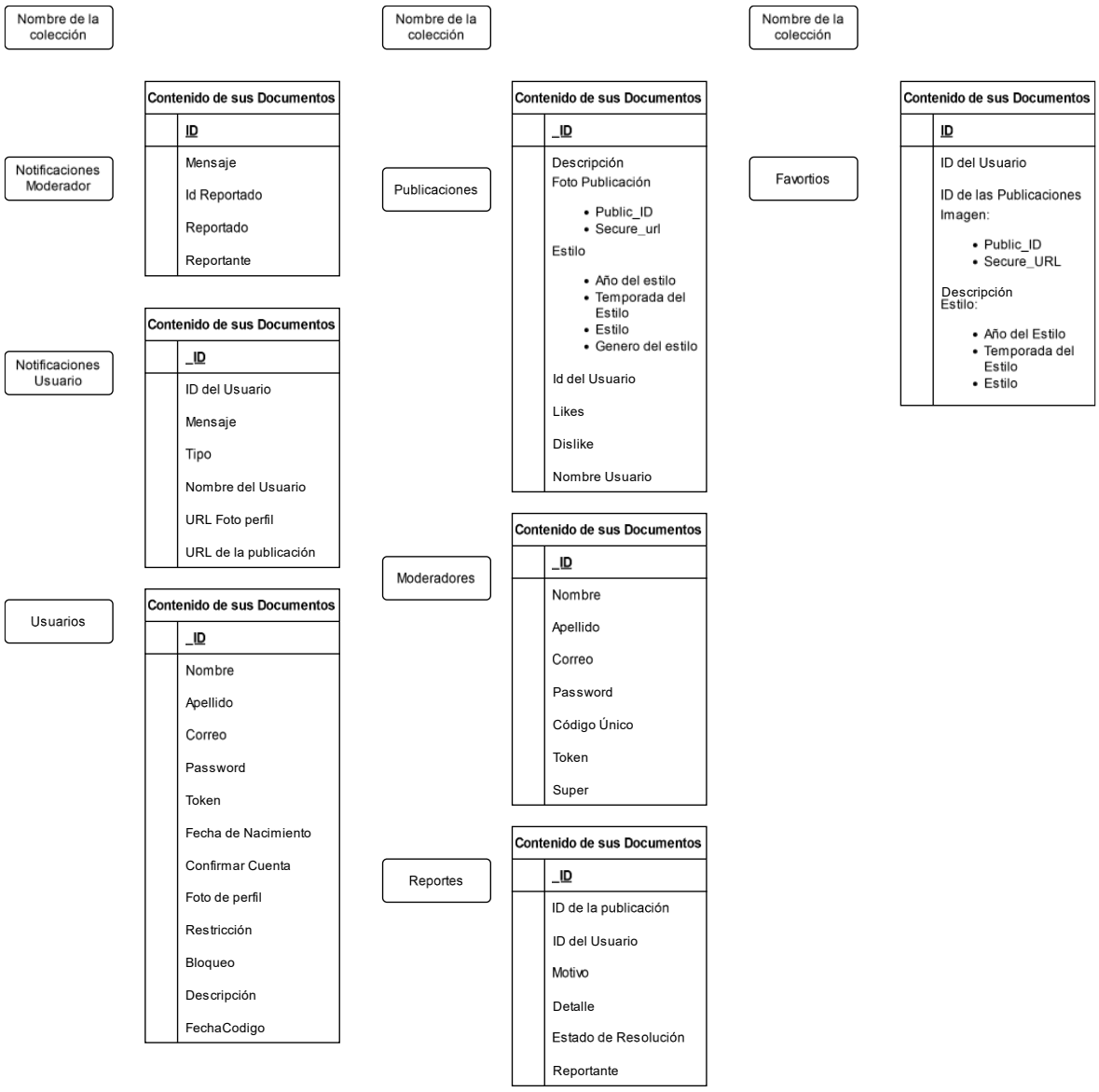


Figura 2.1: Colecciones de la Base de Datos

Patrón arquitectónico

El patrón de arquitectura es un diseño utilizado para el desarrollo de diversas aplicaciones, esto debido a que plantea grandes ventajas a la hora de la implementación, tales como,

potencialización en la facilidad de mantenimiento, reutilización de código, facilidad de correcciones y separación de conceptos [14].

El patrón arquitectónico del presente Trabajo de Integración Curricular es el modelo MVC, basado en tres componentes: Modelo, Vista y Controlado.

- **Modelo:** En este componente se podrá trabajar con los datos que ingresen y salgan de la aplicación, es decir los datos que se contengan dentro de la base de datos.
- **Vista:** Es el componente que permitirá al usuario visualizar la información por medio de una interfaz.
- **Controlador:** Este componente almacena la funcionalidad del sistema, es decir todas las acciones que se soliciten a la aplicación, y trabaja en conjunto con los otros dos componentes Modelo y Vista.

En la **Figura 2.2** se muestra el patrón arquitectónico que posee ROPDAT, y las herramientas que permiten el funcionamiento del componente.

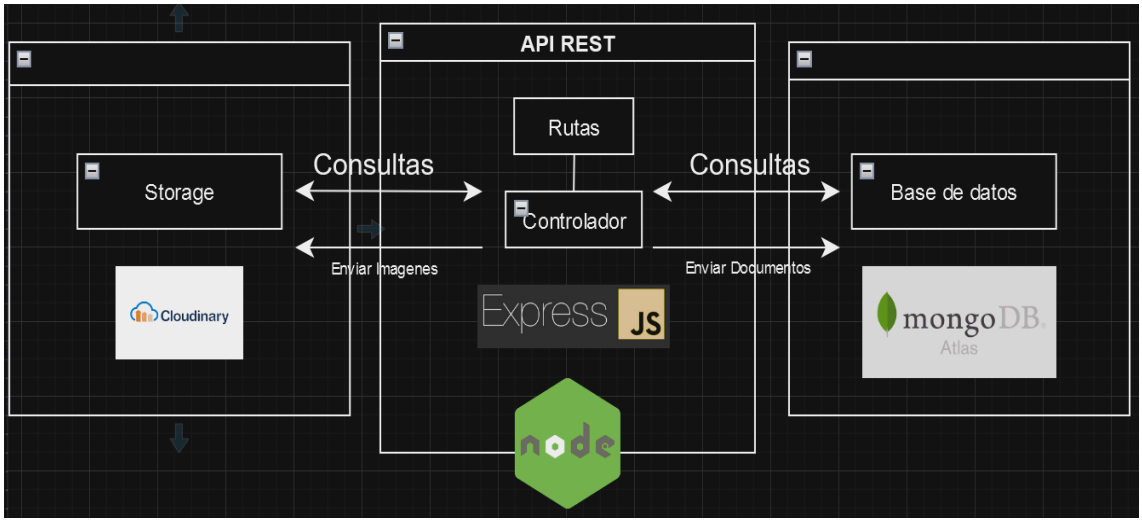


Figura 2.2: Modelo MCV del backend

2.3 Herramientas de desarrollo

En esta sección se detalla las herramientas a utilizar para poder cumplir el modelo arquitectónico, para garantizar el mejor funcionamiento y la eficacia del sistema backend. Todas estas herramientas han permitido la generación de endpoints necesarios para las aplicaciones (móvil y web). La **Tabla 2.3** muestra las herramientas utilizadas dentro del desarrollo del backend.

Tabla 2.3: Herramientas para el desarrollo del Backend

Herramienta	Justificación
Visual Studio Code	Editor de código fuente, permitiendo agregar diferentes funcionalidades que benefician al desarrollador, facilitando esta tarea [15].
Git y Github	Permite el controlar el versiona miento de la aplicación y almacenarlo en un repositorio online dentro de una plataforma de alojamiento [16].
Node js	Entorno de ejecución de JavaScript para el servidor, permitiendo la ejecución de código fuera de navegadores por medio de puertos [6].
Mongo DB atlas	Sistema de gestión de base de datos no relacionales en la nube, otorgando seguridad, disponibilidad y eficiencia para el tráfico de datos entre el servidor y la base de datos [17].
Render	Servicio en línea que permite desplegar la aplicación backend, es decir permite ejecutar el servidor y otro tipo de aplicaciones [18].
Cloudinary	Servicio en línea que permite gestionar medios multimedia (videos y fotos), con la finalidad de no saturar el tráfico de datos a la base de datos [5].
Thunder Client	Es una extensión de VS code, dicha api permite probar los endpoints o APIs desarrolladas [19].
Apache JMeter	Herramienta de prueba de carga para analizar el rendimiento de una aplicación o API [20].

Librerías

A continuación, la **Tabla 2.4:** Librearías usadas**Tabla 2.4** muestra las librerías utilizadas en el desarrollo del backend.

Tabla 2.4: Librearías usadas

Librería	Funcionalidad
Express.js	Es un framework para el desarrollo de aplicaciones web y APIs para el programado en Node.js [21].
Json web token	Librería para generar y comprobar Tokens de acceso [22].
Bcryptjs	Librería usada para encriptar campos, permitiendo comprobar dichos campos sin necesidad de desencriptar [23].
Fs-extra	Librería que permite la gestión de archivos dentro del componente [24].

Moment	Librería usada para la gestión de las fechas, permitiendo configurar formato y realizar operaciones entre fechas [25].
Multer	Librería usada para controlar el flujo de archivos dentro del componente [26].
Nodemailer	Librería que permite el envío de correos electrónicos, mediante el uso de credenciales de Gmail [27].
Cloudinary	Librería usada para gestionar los archivos que son enviados dentro del componente [28].
Mongoose	Librería usada para gestionar las funcionalidades con la base de datos MongoDB Atlas [29].

3 RESULTADOS

A continuación, se muestra los resultados obtenidos en el desarrollo de cada Sprint, demostrando el cumplimiento de los objetivos propuestos para el desarrollo del componente backend ROPDAT.

3.1 Sprint 0. Configuración del ambiente de desarrollo

Para este Sprint se realizaron las siguientes actividades:

- Definición de roles
- Recopilación de requerimientos
- Configuración de herramientas necesarias para el desarrollo
- Diseño de la base de Datos en MongoDB
- Configuración de Cloudinary

Definición de roles

En la **Figura 3.1**, se presenta los roles existentes, a su vez que se detalla las actividades permitidas para cada rol.

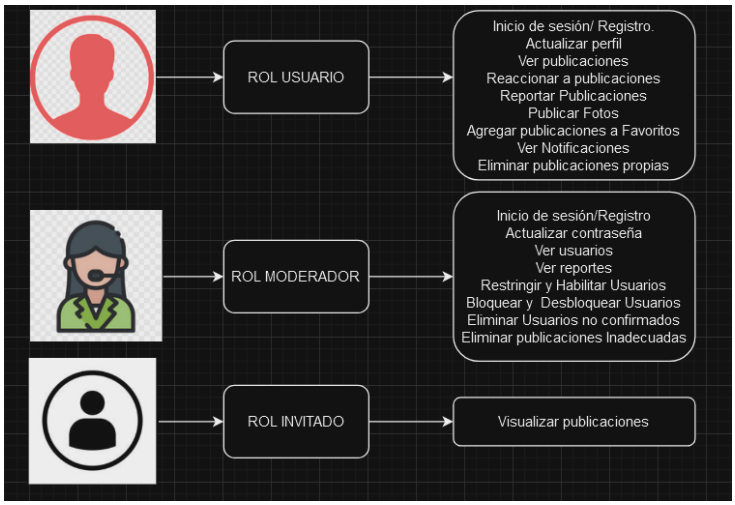


Figura 3.1: Roles del sistema

Recopilación de requerimientos

Endpoints para el registro, login y cambio de contraseña del moderador

- El endpoint de registro permite crear nuevos usuarios, esta funcionalidad es utilizada por la aplicación web puesto que para la parte web, el endpoint será

utilizado por el moderador Super quien es el encargado de agregar nuevos moderadores.

- El endpoint login permite validar las credenciales de los usuarios o moderadores, en caso de ser primer login de los moderadores será necesario cambiar la contraseña.
- El endpoint de cambio de contraseña permite al moderador nuevo cambiar su contraseña de forma obligatoria en el primer login.

Endpoints para la funcionalidad del perfil del usuario

- El endpoint Visualizar perfil, permite al usuario ver su información (nombre, apellido, descripción de la cuenta, foto de perfil), además que permitirá ver las publicaciones que este usuario haya realizado.
- El endpoint actualizar perfil permite modificar la información del usuario (nombre, apellido, descripción).
- El endpoint actualizar contraseña permite actualizar la contraseña al usuario según lo desee
- El endpoint recuperar contraseña, permite cambiar la contraseña al usuario o moderador en caso de que de que sea olvidada.
- El endpoint visualizar favoritos permite visualizar las publicaciones que el usuario haya guardado como favoritas.

Endpoints para la creación de publicaciones

- El endpoint para visualizar publicaciones, será utilizado en la pantalla Home de la aplicación móvil, mostrando las publicaciones existentes realizadas por los usuarios.
- El endpoint publicar, permite al usuario escoger la foto del estilo que desee publicar y agregar información adicional (Descripción, Estilo, Temporada y Época al que pertenece el vestuario, y el Genero al que está dirigido el vestuario) de la publicación realizada.
- El endpoint actualizar información de publicación permite al usuario modificar la información de su publicación en caso de ser necesario.
- El endpoint eliminar, permite al usuario borrar la publicación que desee.

Endpoints para las reacciones del usuario

- Los endpoints para reaccionar permiten al usuario dar like o dislike a una publicación, así también permiten borrar la reacción dada a una publicación.

- El endpoint agregar a favoritos permite guardar la publicación que le agrade al usuario en un apartado específico de favoritos.
- El endpoint borrar de favoritos, elimina la publicación deseada de la lista de favoritos del usuario.
- El endpoint reportar permite al usuario enviar un reporte de una publicación a los moderadores.
- El endpoint buscar permite al usuario filtrar los estilos que desee.

Endpoint para invitados

- El endpoint publicaciones para el rol invitados, permite que la aplicación web muestre las publicaciones de la aplicación móvil, pero sin la posibilidad de interactuar con estas.

Endpoints de moderadores

- El endpoint visualizar usuarios, permite a los moderadores visualizar todos los usuarios registrados dentro de la aplicación móvil.
- El endpoint visualizar reportes permite al moderador visualizar los reportes realizados a las publicaciones.
- Los endpoints bloquear y restringir permiten al moderador sancionar (restringir la cuenta por 3, 7, 15 días o bloquear de forma indefinida) las cuentas de los usuarios que hayan sido reportados.
- Los endpoints para desbloquear y habilitar permiten al moderador retirar la sanción a las cuentas de los usuarios.
- El endpoint borrar publicación elimina la publicación reportada en caso de ser necesario.
- El endpoint eliminar usuario, permite al moderador eliminar las cuentas de los usuarios (que no hayan confirmado su correo después de en cierto periodo de tiempo).

Endpoints para notificaciones

- El endpoint ver notificaciones de usuario permite al usuario ver sus notificaciones tanto de las reacciones obtenidas como alertas de restricciones, por parte del rol moderador este endpoint permite la visualización de las notificaciones de cada reporte realizado en la aplicación móvil.

Configuración de herramientas necesarias para el desarrollo

La configuración de las herramientas se ha iniciado con la instalación de Visual Code Studio y Node.js, además de instalar librerías para el desarrollo de la aplicación, aquellas son visibles en la **Tabla 2.4**. La **Figura 3.2** muestra la estructura del proyecto.

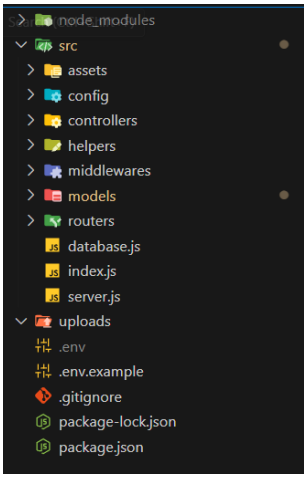


Figura 3.2: Estructura del Proyecto

Diseño de la base de datos en MongoDB atlas

La base de datos no relacional se ha creado en MongoDB atlas, almacenando la información de la aplicación en la nube, El diseño de las colecciones se presenta en la sección **Diseño de la arquitectura** dentro del apartado **Arquitectura de datos** del presente documento**.**

Configuración de Cloudinary

Está herramienta permite el almacenaje de las diversas fotos (de usuarios como de publicaciones) de la aplicación, para lo cual fue necesario crear una cuenta dentro de esta herramienta, esta mismo nos otorga 3 credenciales privadas para su conectividad, las cuales se han agregado dentro de las variables de entorno de ROPDAT y configuradas en un archivo, la función de configuración se muestra en la **Figura 3.3**.

```
//establecer variables de entorno
cloudinary.config({
    cloud_name: process.env.CLOUD_NAME,
    api_key: process.env.API_KEY,
    api_secret: process.env.API_SECRET,
    secure: true
});
```

Figura 3.3: Configuración de las credencias de Cloudinary

3.2 Sprint 1. Autenticación, Registro y perfil de Usuario.

Para el desarrollo del Sprint 1 se incluye las siguientes actividades:

- Endpoints para registro.
- Endpoints para inicio de sesión.
- Endpoints para perfil de usuario.
- Endpoint para visualizar favoritos.

Endpoints para registro

La funcionalidad del registro se ha dividido en dos endpoints diferentes debido a la naturaleza de los componentes que lo consumen (web y móvil). El primero, para la parte móvil, permite el registro de nuevos usuarios mediante la solicitud al método POST y enviando un formulario con datos del usuario, la **Figura 3.4** presenta el resultado (prueba de funcionalidad) de este endpoint.

Por su parte, para la parte web, el endpoint para el registro requiere de un autenticador y validador debido a que el rol moderador (super moderador, usuario creado en la base de datos), es el único que puede registrar nuevos moderadores, de igual manera se realiza mediante el método POST y enviando un formulario con los datos del moderador, en la **Figura 3.5** se muestra el resultado (prueba de funcionalidad) del endpoint. El uso del método POST se debe a que es una inserción de datos a la base.

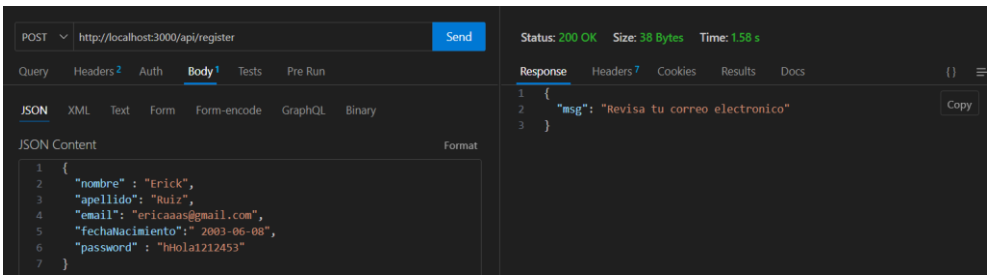


Figura 3.4: Registro de usuarios

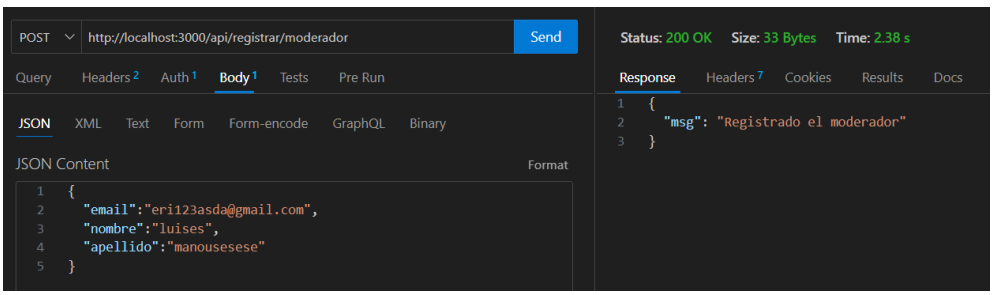


Figura 3.5: Registro de moderadores

Endpoint para inicio de sesión

Estos endpoints tienen por objetivo validar las credenciales tanto del rol Usuario como del rol Moderador, para poder interactuar con sus respectivas funcionalidades, en caso de la parte móvil las credenciales necesarias serán correo y contraseña, una vez iniciado sesión se proporciona un token JWT para autenticar dicho inicio en el resto de endpoints. Para la parte web el inicio de sesión también requiere de las credenciales de correo y contraseña, con la diferencia de que si es el primer inicio de sesión del moderador es necesario cambiar la contraseña por medio del código único, al iniciar se proporciona un token JWT para autenticar el resto de endpoints para el moderador. El uso del método POST en estos endpoints es debido a que solicitamos datos a la base de datos y son comprobados con las credenciales ingresadas.

En la **Figura 3.6** y **Figura 3.7** se presentan los resultados (pruebas de funcionalidad) de inicio de sesión de ambos endpoints de forma exitosa, entregando el token generado para la autenticación del resto de endpoints.

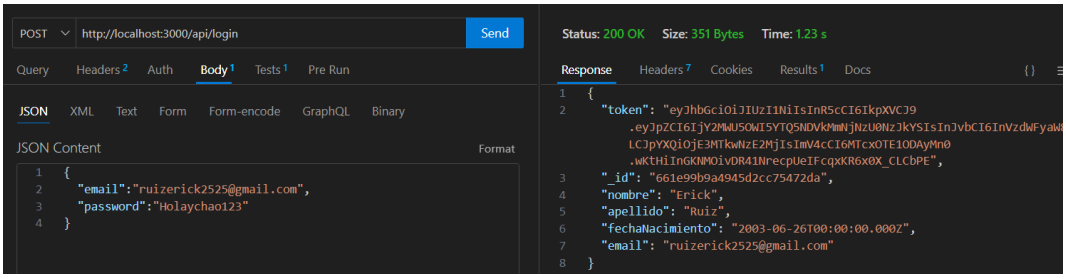


Figura 3.6: Login Usuario

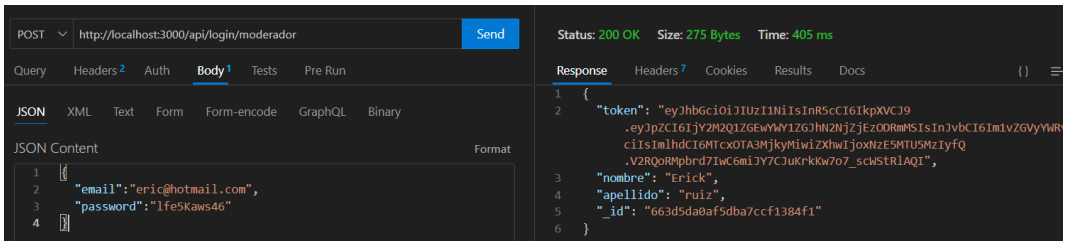


Figura 3.7: Login Moderador

Endpoints para perfil de usuario

Estos endpoints primero validan el inicio de sesión mediante el token generado por el endpoint de inicio de sesión. La primera funcionalidad es visualizar el perfil del usuario, este endpoint mediante el uso del método GET devuelve toda la información del usuario (Nombre, Apellido, Descripción de la Cuenta, Publicaciones Realizadas por el mismo) la **Figura 3.8** muestra el resultado (prueba de funcionalidad) satisfactorio de este endpoint.

La siguiente funcionalidad es actualizar la información (nombre, apellido, foto y descripción) del perfil del usuario, para ello se han desarrollaron dos endpoints, el primer endpoint es actualizar datos, mediante un formulario se actualiza nombre, apellido y descripción del usuario, mientras que el otro endpoint permite sustituir la foto de perfil del usuario, ambos endpoint trabajan mediante el método PUT con la finalidad de actualizar información dentro del servidor, la **Figura 3.9** y la **Figura 3.10**, muestran la correcta funcionalidad (prueba de funcionalidad) de estos endpoints.

El usuario puede cambiar su contraseña en caso de requerirlo por medio de un endpoint específicos para esta tarea; mediante el método PUT y el envío de la contraseña actual y la nueva contraseña el usuario actualiza su contraseña, la **Figura 3.11** presenta la correcta funcionalidad (prueba de funcionalidad) de este endpoint, además, si el usuario olvida su contraseña, puede solicitar el restablecimiento de esta por medio de dos endpoints, el primero solicitando el correo electrónico del usuario para el envío de un correo con un código único, este endpoint utiliza el método POST debido a que toma ciertos datos del usuario, mientras que el segundo endpoint permite al usuario reestablecer su contraseña, esto mediante la validación del código único enviado por correo, dicho endpoint utiliza igualmente el método POST para actualizar dicha contraseña y validar el código único enviado. La **Figura 3.12** y la **Figura 3.13** muestran la funcionalidad (prueba de funcionalidad) de los endpoint mencionados.

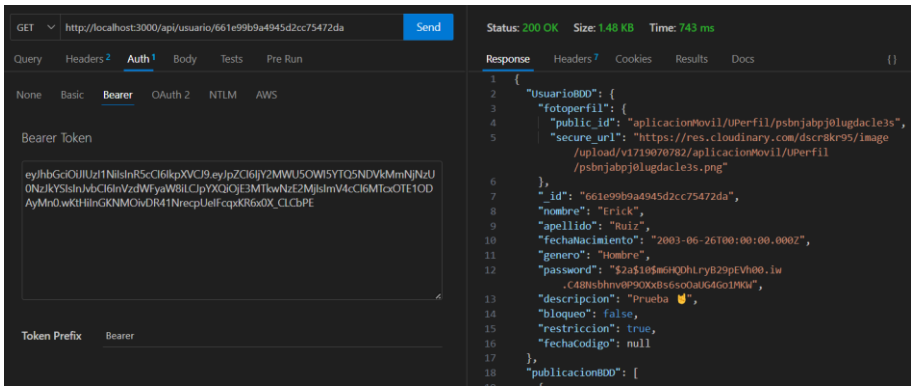


Figura 3.8: Endpoint perfil de usuario

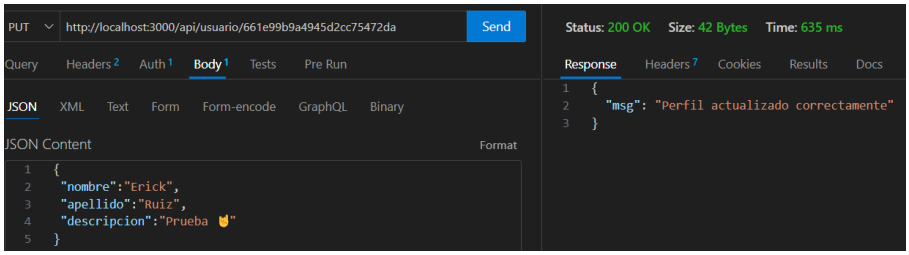


Figura 3.9: Actualización de perfil

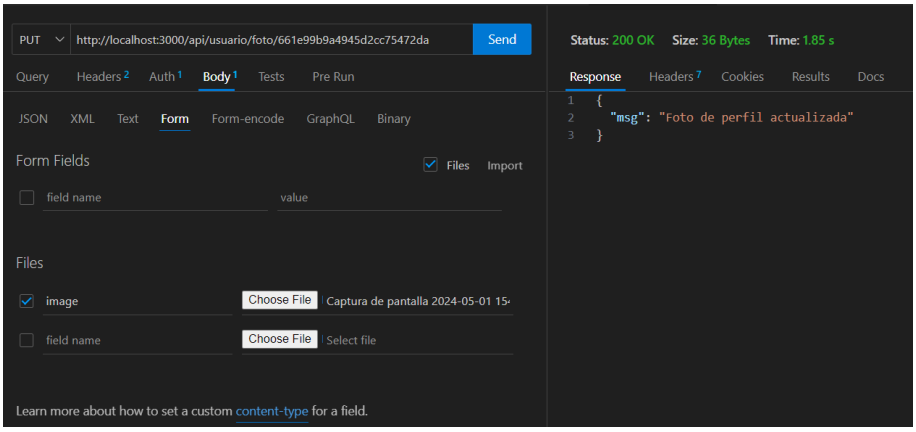


Figura 3.10: Actualización de foto de perfil

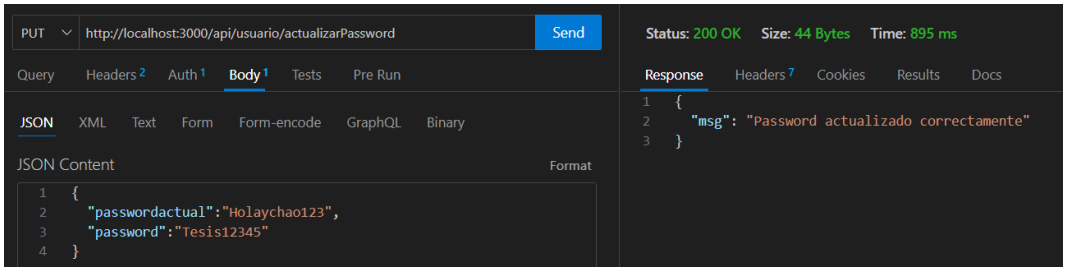


Figura 3.11: Actualizar contraseña

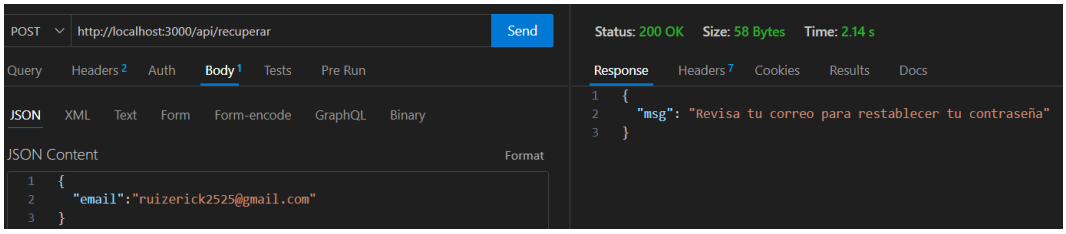


Figura 3.12: Envío del correo de recuperación

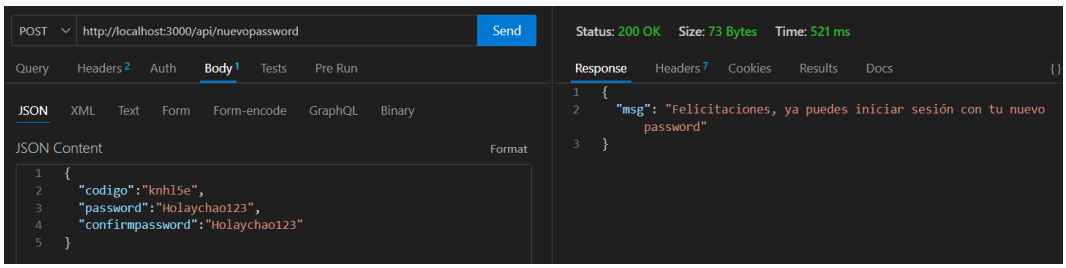


Figura 3.13: Restablecer contraseña

Endpoint para visualizar favoritos

En el perfil del usuario se puede visualizar las publicaciones favoritas, el endpoint mediante el método GET muestra las diversas publicaciones favoritas guardadas por el usuario logueado. La **Figura 3.14** muestra la correcta funcionalidad (pruebas de funcionalidad) de este endpoint.

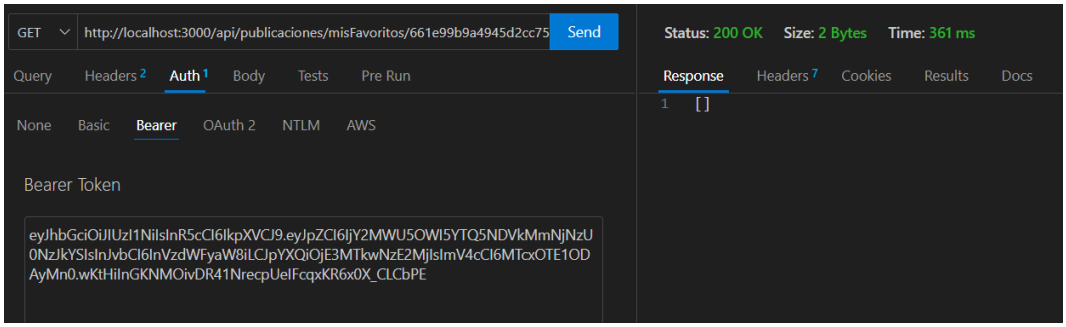


Figura 3.14: Favoritos del del Usuario

3.3 Sprint 2: Módulo de publicaciones e interacciones

El desarrollo de este sprint se ha dividido en las siguientes actividades:

- Endpoints para la funcionalidad del módulo publicaciones del usuario
- Endpoints para la funcionalidad del módulo de interacciones

Endpoints para la funcionalidad del módulo de publicaciones del usuario

Este módulo presenta una de las partes más importantes de la aplicación, los endpoints a continuación muestran un CRUD para las publicaciones que el usuario realiza dentro de la aplicación, el primer endpoint es el de publicar que mediante el método POST se creara una nueva publicación, almacenando la imagen en la herramienta Cloudinary con la finalidad de evitar fallos en la base de datos. La **Figura 3.15** muestra el resultado (prueba de funcionalidad) del endpoint.

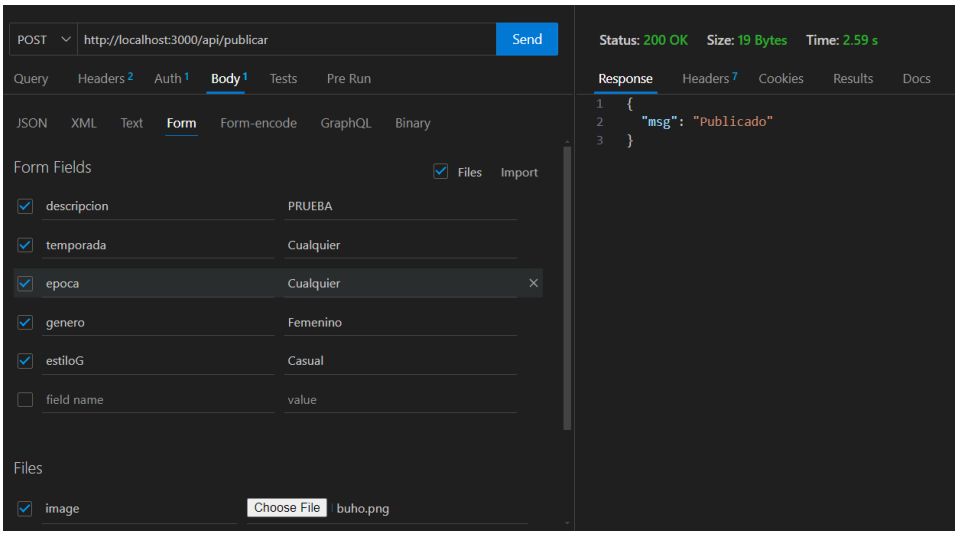


Figura 3.15: Endpoint publicar

El segundo endpoint permite actualizar una publicación, modificando la información de esta (descripción, temporada, época, genero, estilo), mediante el uso del método PUT dado que se modifica la información en la base de datos. La **Figura 3.16** muestra la funcionalidad (prueba de funcionalidad) correcta del endpoint.

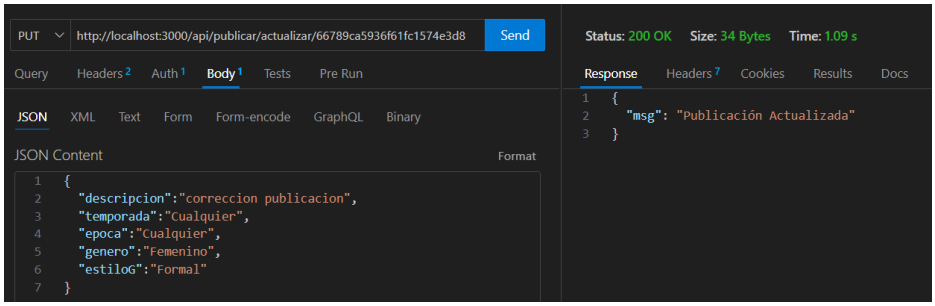


Figura 3.16: Endpoint Actualizar Publicación

Se han desarrollaron dos endpoints para visualizar las publicaciones existentes, uno para visualizar todas las publicaciones en general, y otro para presentar la información de una publicación en específico, la **Figura 3.17** y la **Figura 3.18** muestran el resultado (prueba de funcionalidad) de ambos endpoints, para esta funcionalidad se ha trabajado con el método GET mismo que devuelve información en función a un parámetro específico.

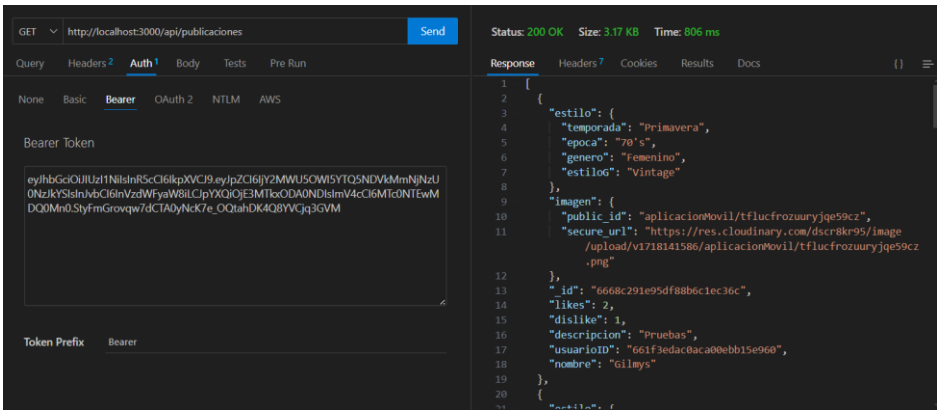


Figura 3.17: Endpoint Todas las Publicaciones

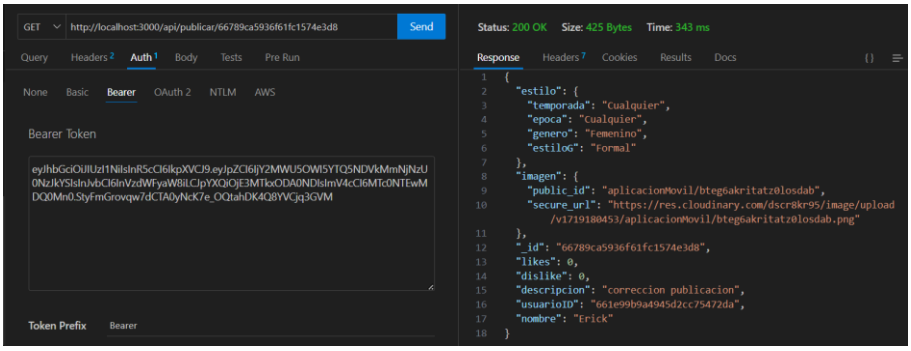


Figura 3.18: Endpoint ver Publicación

Por último, el endpoint eliminar publicación se encarga de borrar una publicación en específico en caso de que el usuario así lo desee. La petición se realiza mediante el método DELETE, debido a que se elimina datos de la base y de Cloudinary. La **Figura 3.19** demuestra el correcto funcionamiento (prueba de funcionalidad) de este endpoint.

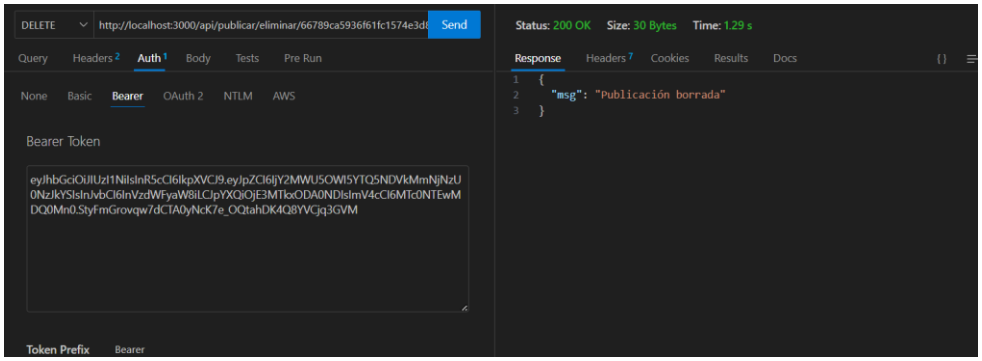


Figura 3.19: Eliminar Publicación

Endpoints para la función de módulo de interacciones

El módulo de interacciones se divide en diversos endpoints, para agregar y eliminar like, dislike y favoritos, además para realizar búsquedas por filtros y, por último, realizar reportes en caso de ser necesario.

Los endpoints para dar like y dislike permiten al usuario reaccionar a las publicaciones de otros usuarios, esto mediante el método PUT. La **Figura 3.20** muestra la funcionalidad (prueba de funcionalidad) de ambos endpoints.

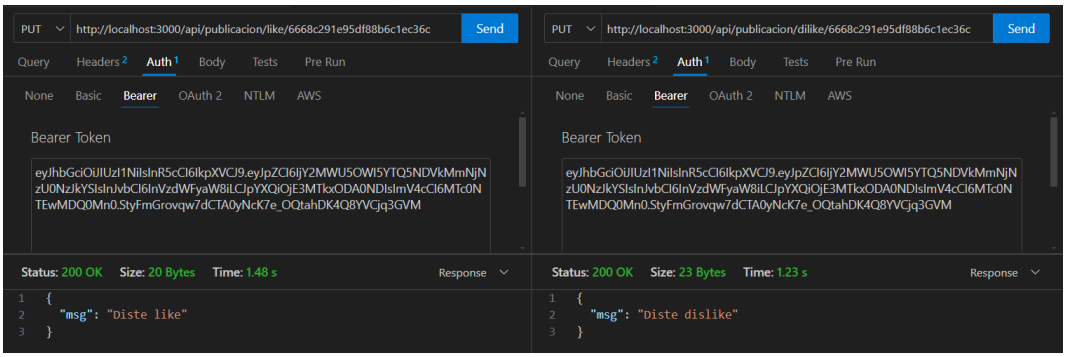


Figura 3.20: Reacciones de las publicaciones

Los endpoints para borrar Like y Dislike, permite al usuario borrar las reacciones que el haya dado a otras publicaciones, de igual manera se utiliza el método PUT en la petición. La **Figura 3.21** muestra la funcionalidad (prueba de funcionalidad) de los dos endpoints.

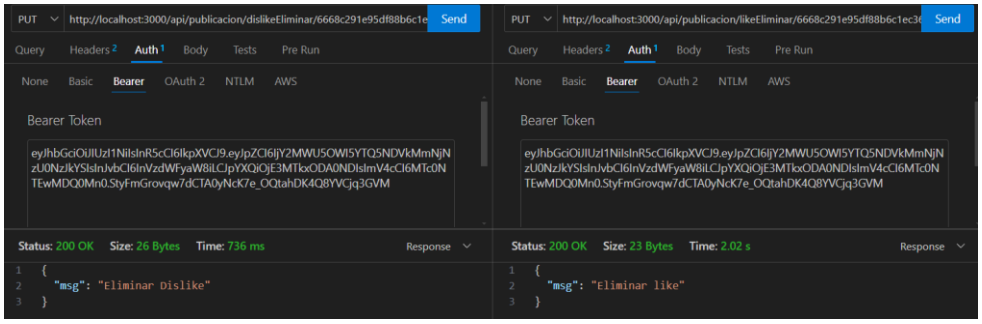


Figura 3.21: Eliminar Reacciones

El endpoint agregar a favorito permite al usuario almacenar una publicación en su lista de Favoritos, esta petición se realiza mediante el uso del método POST, además se cuenta con un endpoint para eliminar aquellas publicaciones favoritas, dicha petición es realizada con el método DELETE. La **Figura 3.22** y **Figura 3.23** la muestran la funcionalidad (prueba de funcionalidad) de estos dos endpoints.

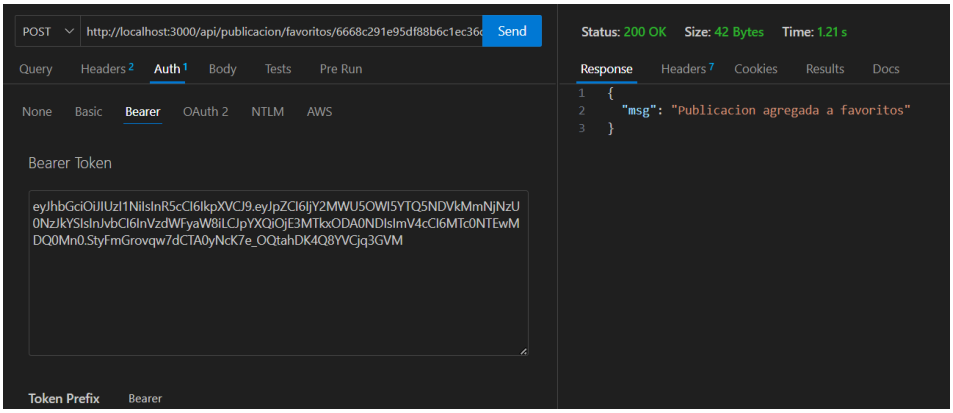


Figura 3.22: Agregar a Favoritos

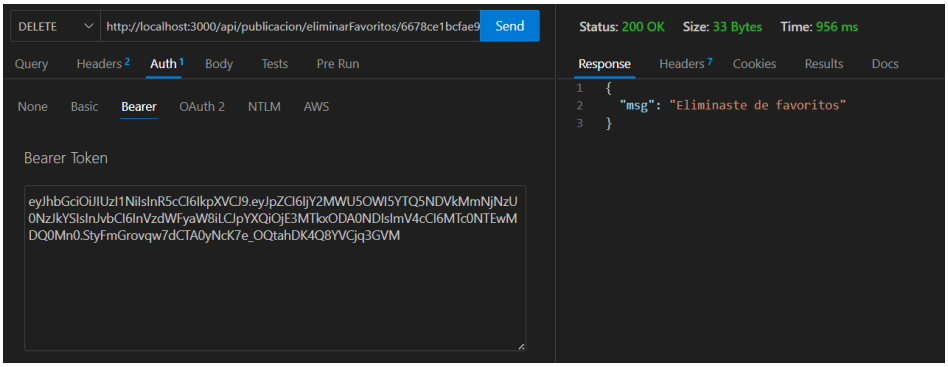


Figura 3.23: Eliminar de Favoritos

El último endpoint de interacción de los usuarios es la búsqueda mediante filtros, la petición se realiza mediante el método POST, esto debido a se envía el filtro por medio de un

formulario, el usuario podrá seleccionar los filtros (Por temporadas, épocas, estilos y/o genero) según su interés. La **Figura 3.24** muestra la correcta funcionalidad (prueba de funcionalidad) de este endpoint.

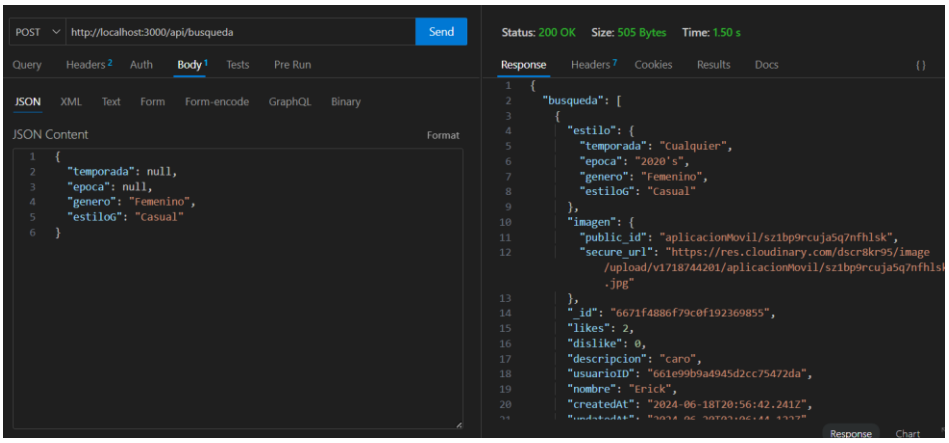


Figura 3.24: Búsqueda de Publicaciones

3.4 Sprint 3: Módulo de moderadores

El desarrollo de este sprint se ha dividido en las siguientes actividades:

- Endpoints visualizar usuarios
- Endpoints para restablecer contraseña del moderador
- Endpoints Visualizar reportes
- Endpoint Bloquear Usuarios
- Endpoints Resolución de reportes
- Endpoint Restringir
- Endpoints para desbloquear y habilitar usuarios
- Endpoint Eliminar Usuario no confirmado
- Endpoint Eliminar Moderadores
- Endpoint Visualizar Moderadores confirmados
- Endpoint Visualizar Moderadores no confirmados

Endpoint visualizar usuarios

Los endpoints visualizar usuarios permiten al moderador visualizar una lista de usuarios registrados, dichos listados presentan información del usuario (nombre, apellido, fecha de registro, confirmación del correo), el primer endpoint presenta a los usuarios que han confirmado su correo, mientras que el segundo endpoint presenta el listado de los usuarios

que aún no han confirmado su correo, ambos endpoints realizan peticiones mediante el método GET, dado que devuelve información de la base de datos. La **Figura 3.25** y **Figura 3.26** presentan la funcionalidad (prueba de funcionalidad) correcta de estos endpoints.

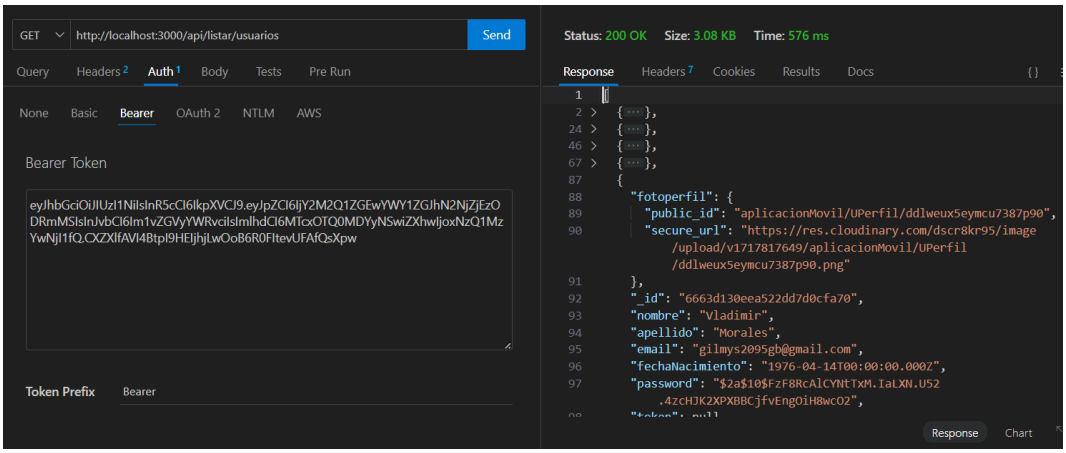


Figura 3.25: Listar usuarios

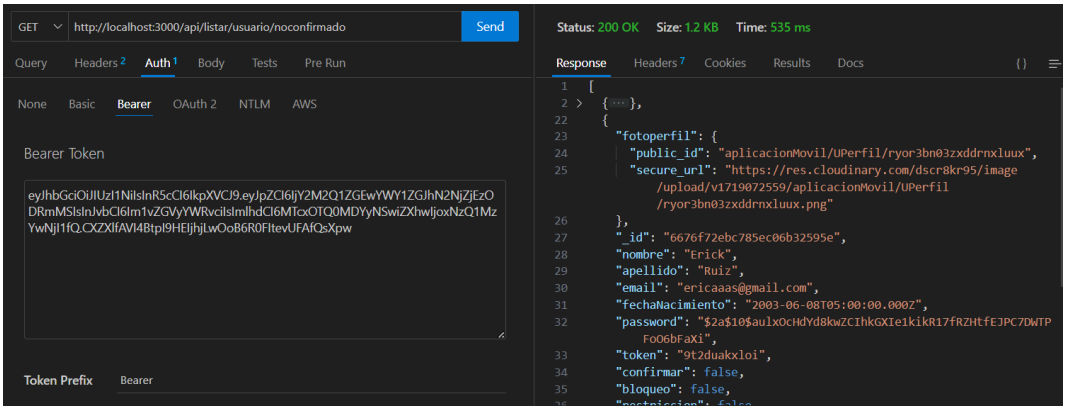


Figura 3.26: Listar usuarios sin confirmar

Endpoints para restablecer contraseña del moderador

El proceso de restablecer contraseña se divide en tres endpoints, el primer endpoint mediante el uso del método POST permite el envío de un correo de recuperación al correo electrónico del moderador, además de generar un token y almacenarlo en la base de datos, el segundo endpoint se encarga de verificar dicho endpoint esto mediante el método GET, dado que el token es un parámetro enviado por el URL del endpoint, el último endpoint se encarga de restablecer la contraseña del moderador, solicitando el ingreso de la nueva contraseña, el método a utilizar en esta petición es POST. La **Figura 3.27**, **Figura 3.28** y la **Figura 3.29** muestran la funcionalidad (prueba de funcionalidad) correcta de estos endpoints.

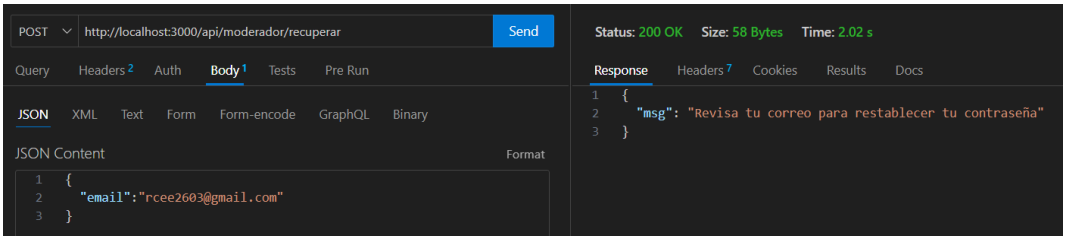


Figura 3.27: Solicitar restablecer contraseña

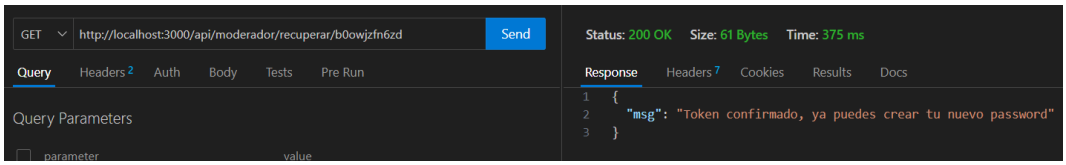


Figura 3.28: Confirmar token enviado

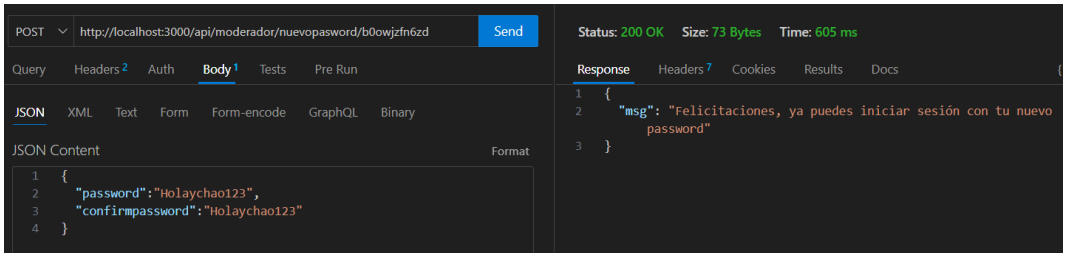


Figura 3.29: Cambiar la contraseña

Endpoints para visualizar reportes

El endpoint visualizar reportes permite al moderador visualizar todos los reportes realizados en la aplicación móvil, por otro lado, se ha desarrollado otro endpoint que permite al moderador visualizar un reporte en específico, las peticiones son realizadas con el método GET, esto debido a que devuelve información de la base. La **Figura 3.30** y la **Figura 3.31** presentan la correcta ejecución (prueba de funcionalidad) de estos endpoints.

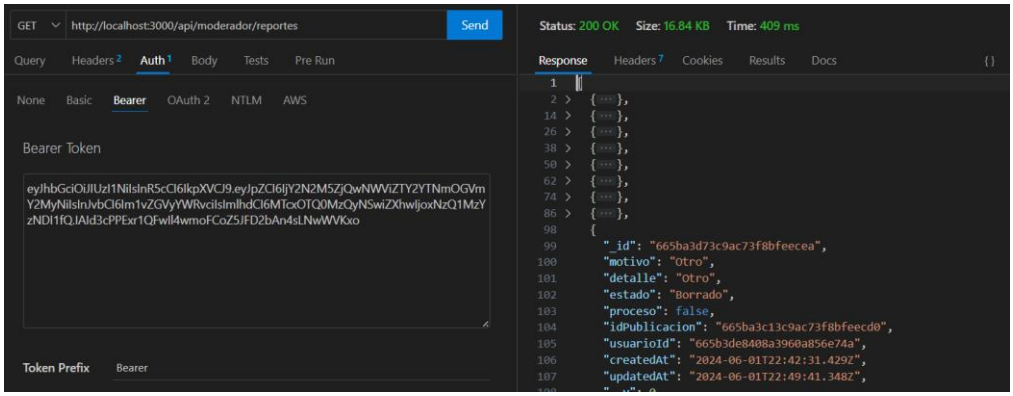


Figura 3.30: Visualizar reportes

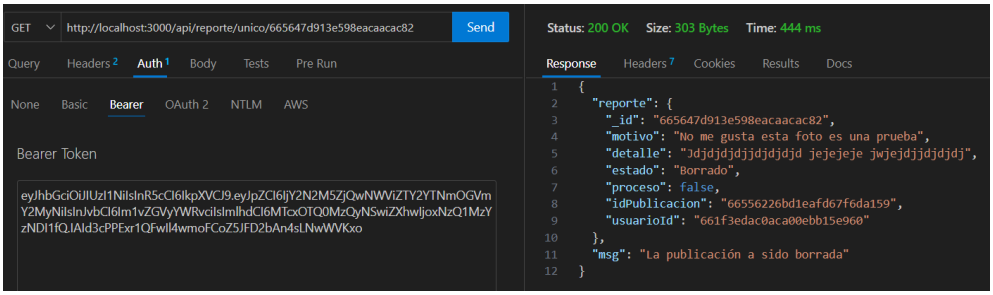


Figura 3.31: Visualizar reporte único

Endpoint bloquear Usuario

El endpoint bloquear Usuario permite al moderador llevar a cabo la última sanción a un usuario reportado, la cual consiste en bloquear de forma permanente la cuenta del usuario, el método utilizado es PUT, debido a que se actualizara un valor booleano dentro de la base de datos. La **Figura 3.32** muestra el correcto funcionamiento (prueba de funcionalidad) de este endpoint.

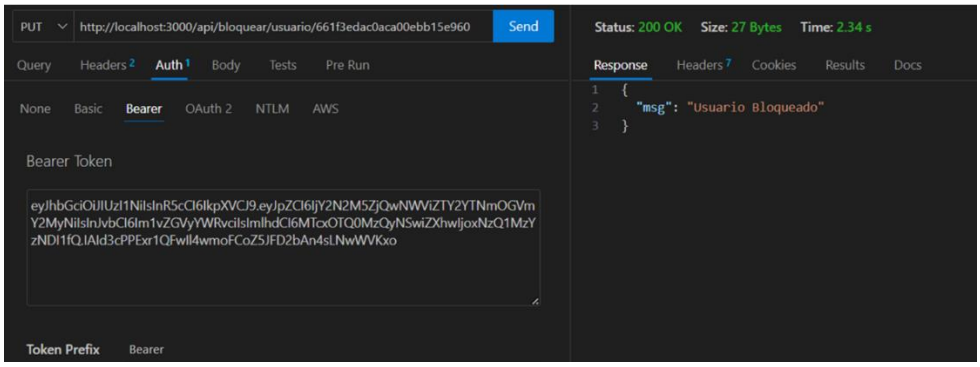


Figura 3.32: Bloquear Usuario

Endpoint Restringir Usuario

El endpoint restringir usuario, permite al moderador sancionar al usuario reportado, el tiempo de restricción es de 3, 7 y 15 días dependiendo de las veces que haya sido sancionado, la petición se realiza por medio del método PUT, dado que se modifica un valor booleano dentro de la base de datos. La **Figura 3.33** muestra el correcto funcionamiento (prueba de funcionalidad) de este endpoint.

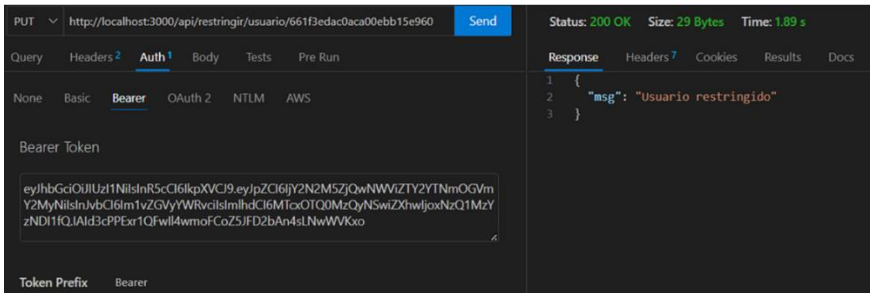


Figura 3.33: Restringir Usuario

Endpoints Resolución de Reportes

El endpoint borrar publicación, permite al moderador eliminar publicaciones reportadas según su resolución al reporte recibido, la petición se realiza por medio del método DELETE, ya que se borra información de la base de datos. El endpoint reporte falso permite al moderador solucionar un reporte en caso de que determine que es un reporte falso, la petición se realiza con el método PUT dado a que se cambia el estado del reporte. El endpoint cambio de estado, permite cambiar el estado de los reportes no resueltos de una publicación que ha sido eliminada, el estado del reporte pasa a ser “Borrado”, la petición es realizada con el método PUT. La **Figura 3.34**, **Figura 3.35** y la **Figura 3.36** presenta la correcta funcionalidad (prueba de funcionanalidad) de estos endpoints.

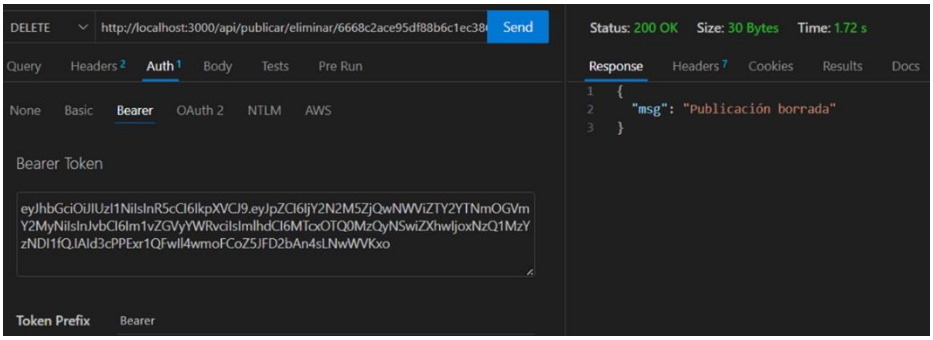


Figura 3.34: Eliminación de Publicación

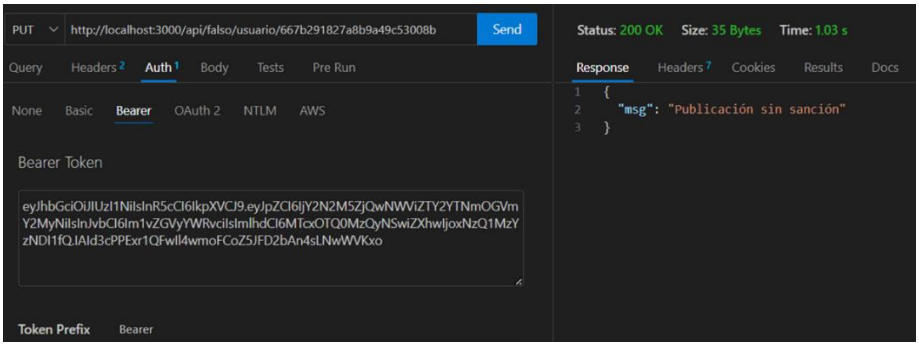


Figura 3.35: Resolución de reporte falso

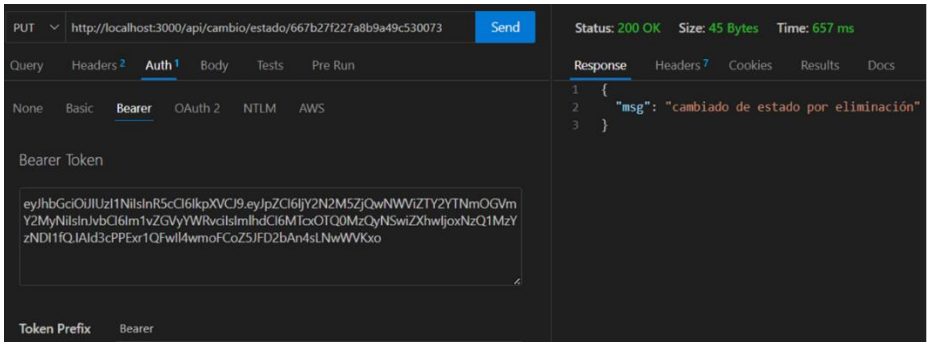


Figura 3.36: Cambio de estado

Endpoints para Desbloquear y Habilitar usuarios

El endpoint Habilitar cuenta permite al moderador retirar la sanción impuesta al usuario, por su parte, el endpoint Desbloquear usuarios permite al moderador retirar el bloqueo a la cuenta de un usuario, ambas peticiones se realizan por medio del método PUT dado que se modifica una variable booleana de la base de datos. La **Figura 3.37** y la **Figura 3.38** muestran el correcto funcionamiento (prueba de funcionalidad) de estos endpoints.

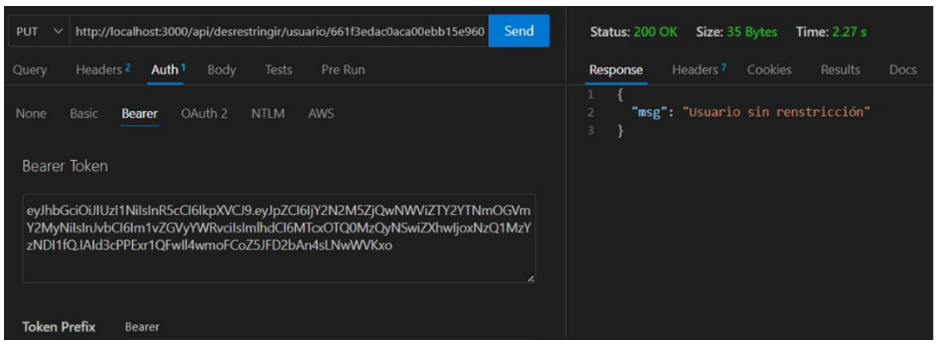


Figura 3.37: Habilitar Usuario

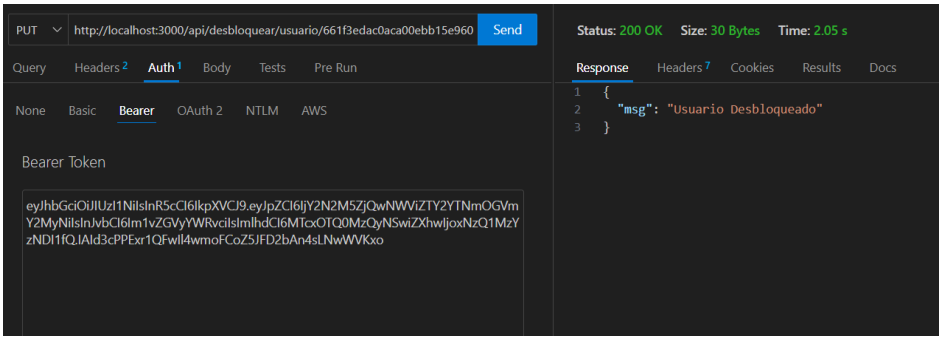


Figura 3.38: Desbloquear Usuario

Endpoint eliminar usuario no confirmado

El endpoint eliminar usuario no confirmado permite al moderador eliminar aquellas cuentas de usuario que no han confirmado su correo electrónico en un periodo de tiempo (5 días), la petición utilizada es DELETE debido a la eliminación de cuentas de la base de datos. La **Figura 3.39** presenta la funcionalidad (prueba de funcionalidad) de este endpoint.

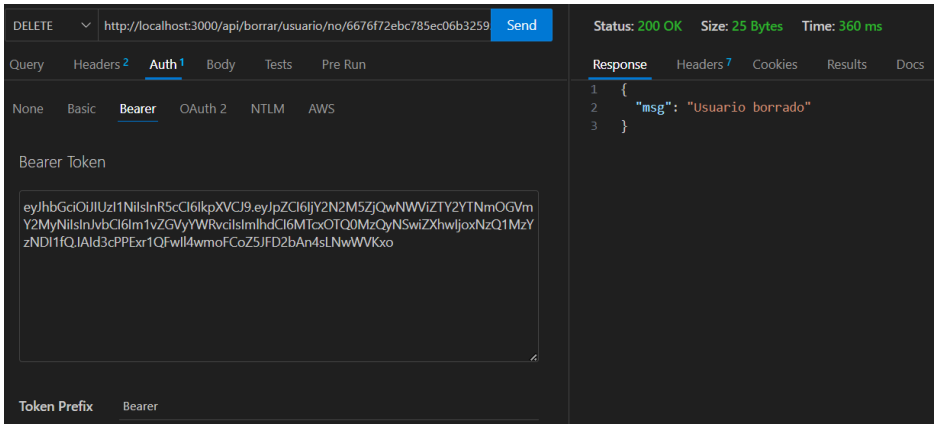


Figura 3.39: Eliminar Usuario no confirmado

Endpoint Eliminar Moderador

El endpoint eliminar moderador permite al super moderador (moderador creado en la base de datos) eliminar moderadores que han confirmado su correo y que no hayan confirmado su correo, el método utilizado es DELETE debido a la eliminación de información de la base de datos. La **Figura 3.40** presenta la funcionalidad (prueba de funcionalidad) de este endpoint.

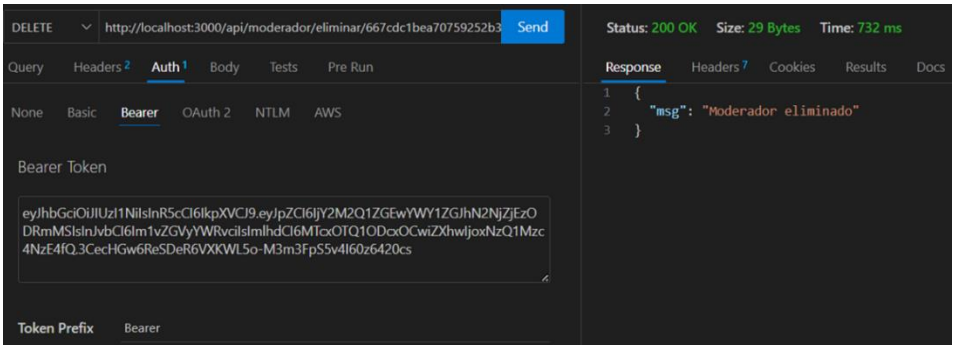


Figura 3.40: Eliminar Moderador

Endpoint Visualizar moderadores Confirmados

El endpoint visualizar moderadores confirmados permite al Super Moderador visualizar todos lo moderadores que han ingresado por primera vez a la aplicación, la cuenta de un moderador es confirmada cuando el código único es utilizado, mismo que es anulado en dicho primer inicio de sesión. La petición se realiza con el método GET dado que se solicita información a la base de datos. La **Figura 3.41** presenta la correcta funcionalidad (prueba de funcionalidad) de este endpoint.

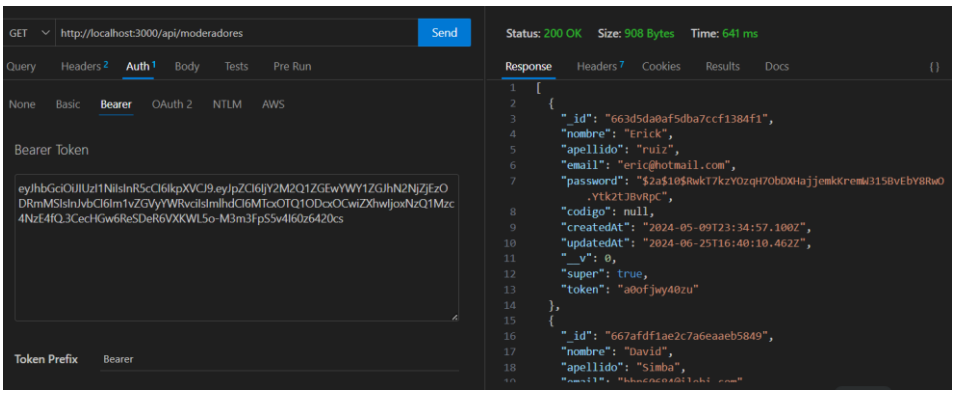


Figura 3.41: Visualizar Moderadores confirmados

Endpoint visualizar Moderadores no confirmados

El endpoint visualizar moderadores no confirmados permite al super moderador visualizar aquellas cuentas que aun poseen el código único, mismo que debe ser nulo cuando un moderador inicia sesión por primera vez, la petición se realiza con el método GET dado que se solicita información a la base de datos. La **Figura 3.42** muestra la correcta funcionalidad (prueba de funcionalidad) de este endpoint.

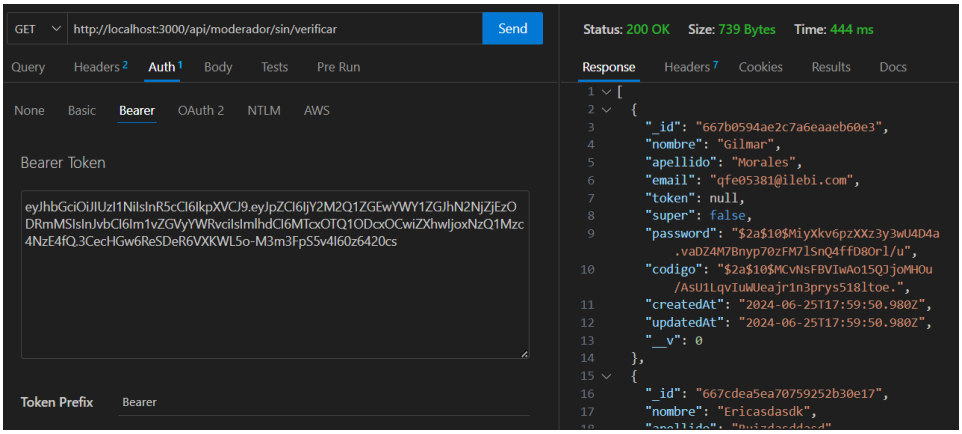


Figura 3.42: Visualizar moderadores no confirmados

3.5 Sprint 4: Módulo Invitado y Notificaciones

Las actividades por cumplir en este sprint son las siguientes:

- Endpoint para el invitado
- Endpoint para notificaciones de Usuario
- Endpoints para notificaciones de Moderador

Endpoint para el invitado

El endpoint para el invitado permite al rol invitado visualizar todas las publicaciones realizadas, con la diferencia de que el invitado no puede interactuar con las publicaciones, el método para realizar esta petición es GET ya que se solicita información de la base de datos, además este endpoint no solicita ningún autenticador para poder utilizarlo. La **Figura 3.43** presenta la funcionalidad (prueba de funcionalidad) correcta de este endpoint.

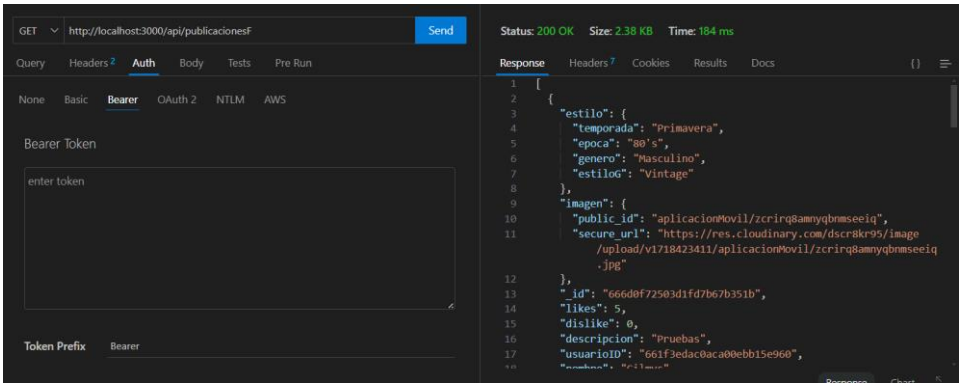


Figura 3.43: Endpoint del Invitado

Endpoint para notificaciones de Usuario

Las notificaciones del usuario se realizan por medio de funciones dentro de los endpoints dar like y restringir Usuario. El endpoint para notificaciones de Usuario permite al usuario visualizar todas las notificaciones de su cuenta, estas notificaciones se dividen en dos tipos: Informativa y de Alerta, la petición se realiza mediante el método GET, debido a que se solicita información a la base de datos de un usuario en específico. La **Figura 3.44** presenta la correcta funcionalidad (prueba de funcionalidad) de este endpoint. La **Figura 3.45** presenta un ejemplo de creación de notificación al momento de utilizar el endpoint dar like.

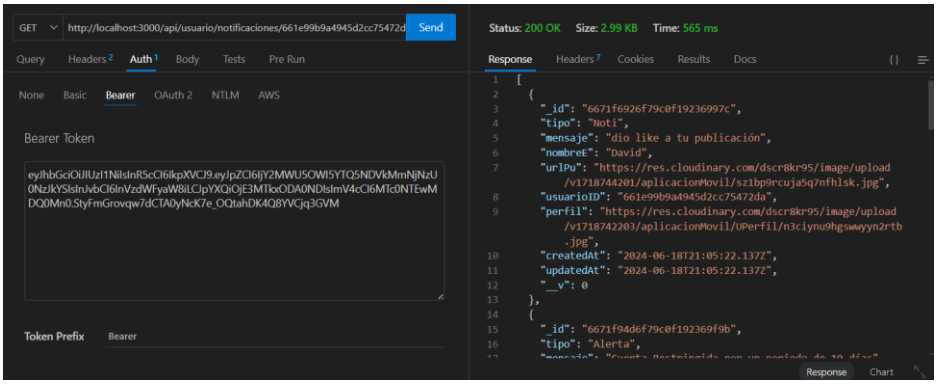


Figura 3.44: Notificaciones de Usuario

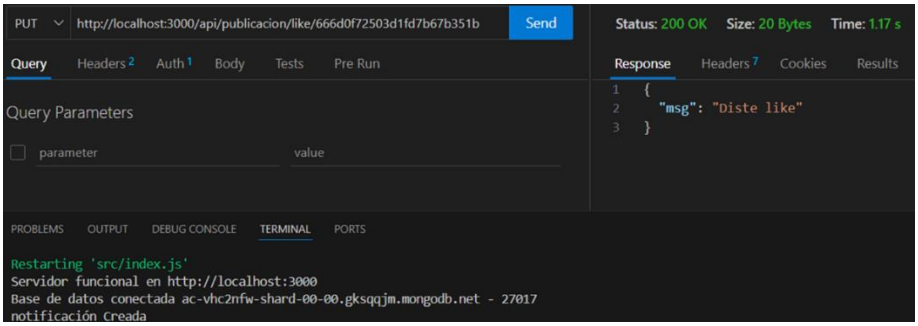


Figura 3.45: Ejemplo de creación de Notificación

Endpoint notificaciones de moderadores

Las notificaciones de moderador se realizan al momento utilizar el endpoint reportar publicación, el endpoint notificaciones de moderadores permite a los moderadores visualizar las notificaciones de reportes realizados, la petición se realiza con el método GET, dado que se solicita información de la base de datos. La **Figura 3.46** presenta la funcionalidad (prueba de funcionalidad) de este endpoint.

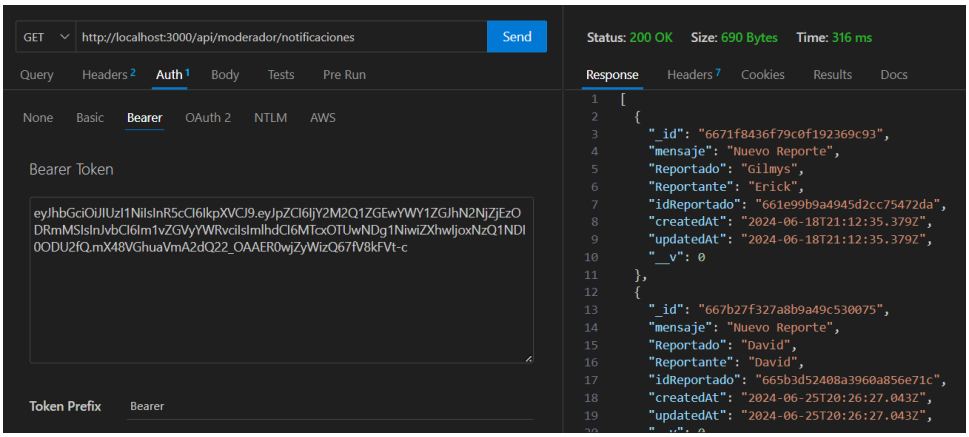


Figura 3.46: Notificaciones de Moderador

3.6 Sprint 5: Pruebas y Despliegue

El desarrollo de este Sprint incluye las siguientes actividades:

- Pruebas de Funcionalidad
- Pruebas de Carga
- Pruebas de Rendimiento
- Despliegue API Rest en Render

Pruebas de Funcionalidad

Las pruebas de Funcionalidad son aquellas realizadas a los módulos o endpoints que buscan verificar la correcta funcionalidad de las características del software planeadas en el desarrollo. Este tipo de pruebas se realizan verificando la funcionalidad del endpoint proporcionándole datos de entrada y capturando los datos de salida para posterior analizar si el resultado es el esperado [30].

La **Figura 3.47** y **Figura 3.48** presentan la prueba de funcionalidad al endpoint listar usuario no verificado, presentando los datos recibidos de esta petición, mostrando su correcta funcionalidad. En **Pruebas de Funcionalidad** se presentan todas las pruebas de funcionalidad realizadas al componente.

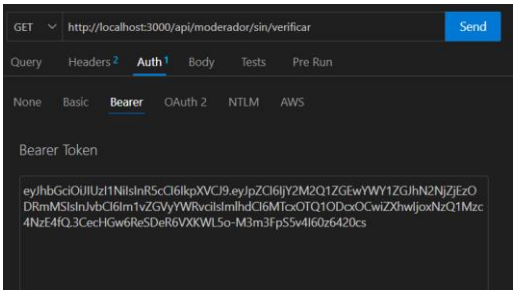


Figura 3.47: Prueba de Funcionalidad Listar usuarios no verificados

Figura 3.48: Resultado de la Prueba de Funcionalidad

Prueba de carga

La prueba de carga permite determinar el comportamiento de la aplicación en diversas condiciones de carga, y determinar el punto de ruptura de la aplicación, que permitiendo analizar y verificar el rendimiento de la aplicación según la carga prevista. Las pruebas de carga buscan simular situaciones de carga realistas y proporcionar información acerca del comportamiento de la aplicación, permitiendo verificar posibles fallos ante cargas amplias o máximas [31].

Para realizar las pruebas de carga se ha empleado Apache JMeter, esta herramienta permite crear y configurar diferentes simulaciones para comprobar el rendimiento del componente. La **Tabla 3.1** muestra la capacidad resultante de las peticiones HTTPS realizadas a ROPDAT.

Tabla 3.1: Ejecución de las pruebas de carga para 23 endpoints de ROPDAT

Cantidad de peticiones	Tiempo de respuesta (Media)	Detalle
1150	00:00:08.01	Funcional

Los resultados obtenidos de las pruebas de carga presentan la funcionalidad correcta de las peticiones realizadas a la API Rest ROPDAT para usuarios finales. No se observan fallos en los tiempos de ejecución, además de procesar todas las peticiones de forma correcta. En el **PRUEBA DE CARGA** se visualiza las evidencias de las pruebas de carga realizadas.

Pruebas de rendimiento

Las pruebas de rendimiento permiten determinar la velocidad, estabilidad y capacidad de una aplicación, evalúan el rendimiento de la aplicación sometiéndola a varias cargas de trabajo donde se simula un entorno real, las pruebas de rendimiento permiten analizar la velocidad de procesamiento y transferencia de datos, tiempo de respuesta y posibles fallos al aplicar cargas pesadas [32].

La **Figura 3.49** presenta el rendimiento del componente ROPDAT al ser sometido a una carga de 1150 peticiones HTTPS a diversos endpoints del componente mostrando un tiempo de ejecución de 56.956 segundos, y una media de envió de datos de 36673.3 bytes, lo uqe significa que tiene buen rendimiento. En el **PRUEBAS DE RENDIMIENTO** se muestra la prueba de rendimiento realizada al componente.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Login	50	22683	4569	41989	11731.22	0.00%	1.1/sec	0.84	0.26	806.0
perfil Usuario	50	13091	3419	40710	11116.61	0.00%	57.6/min	79.03	0.38	84337.0
Actualizar perfil	50	5195	1100	35140	6082.04	0.00%	1.1/sec	0.53	0.59	495.0
actualizar Foto ...	50	13742	4680	28112	5717.48	0.00%	1.0/sec	0.49	59.15	489.0
notificaciones U	50	4236	1581	12099	2548.26	0.00%	1.4/sec	5.34	0.56	3986.0
Publicaciones U	50	3626	1899	9698	1605.17	0.00%	1.4/sec	100.21	0.52	73695.4
Publicaciones I	50	2358	898	5317	1094.05	0.00%	1.5/sec	107.42	0.25	74786.0
Publicar	50	10822	3836	20002	3833.83	0.00%	1.4/sec	0.65	1116.92	472.0
Actualizar contr...	50	36265	9599	56956	15049.61	0.00%	48.7/min	0.39	0.38	497.0
Actualizar Public...	50	1728	941	3912	793.01	0.00%	5.9/sec	2.79	3.35	487.0
Ver publicacion	50	1469	573	3191	842.81	0.00%	5.0/sec	4.20	1.98	861.0
Dar like	50	2312	1307	5007	841.85	0.00%	3.9/sec	1.81	1.66	473.0
Dar dislike	50	2678	1390	4390	836.71	0.00%	3.9/sec	1.83	1.67	476.0
Reporte Nuevo	50	3039	1505	6194	1056.33	0.00%	4.2/sec	1.96	1.99	478.0
Ver favoritos	50	2038	667	6222	1145.03	0.00%	4.5/sec	8.60	1.88	1941.0
Busqueda	50	3595	1045	9304	2018.67	0.00%	3.4/sec	356.58	0.91	108621.0
Ver usuario Conf...	50	3476	772	9566	2415.56	0.00%	2.2/sec	7.59	0.82	3607.0
Resportes realiz...	50	7377	1816	16098	3783.72	0.00%	1.3/sec	101.27	0.51	78001.0
notificaciones m...	50	4910	1177	15104	3586.38	0.00%	1.3/sec	1.12	0.49	915.0
listar usuario no ...	50	10380	3371	18370	5206.19	0.00%	1.2/sec	412.11	0.47	350898.5
Listar moderado...	50	6248	845	16338	4456.52	0.00%	1.3/sec	1.72	0.49	1363.0
moderador sin v...	50	6467	1288	18478	4150.95	0.00%	1.3/sec	70.54	0.51	55072.1
Login moderador	50	16636	3912	27899	7216.35	0.00%	1.5/sec	1.09	0.38	730.0
Total	1150	8016	573	56956	9854.39	0.00%	1.8/sec	64.97	68.13	36673.3

Figura 3.49: Pruebas de rendimiento al componente ROPDAT

Despliegue API Rest en Render

El objetivo del despliegue del backend ROPDAT es facilitar el consumo exitoso del componente por parte las aplicaciones web y móvil. El despliegue del componente se realiza por medio del servicio web Render, la base de datos en la nube se encuentra en MongoDB atlas y el almacenaje de archivos en Cloudinary. La **Figura 3.50** muestra el despliegue exitoso del servicio web, conectado al repositorio principal del componente en GitHub, mientras que las **Figura 3.51** y **Figura 3.52** presentan la conexión del componente a la base de datos y almacenamiento en la nube.

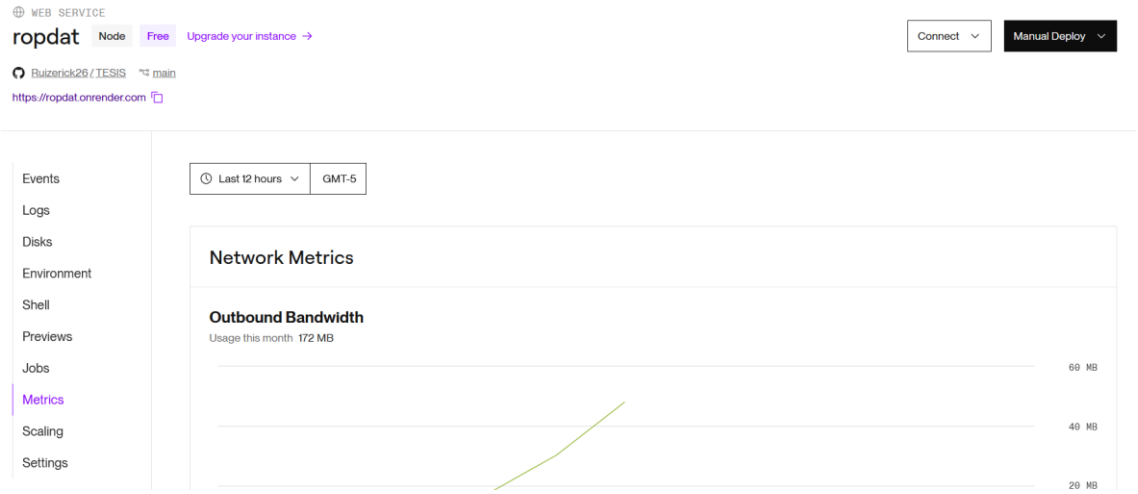


Figura 3.50: Despliegue API Rest ROPDAT

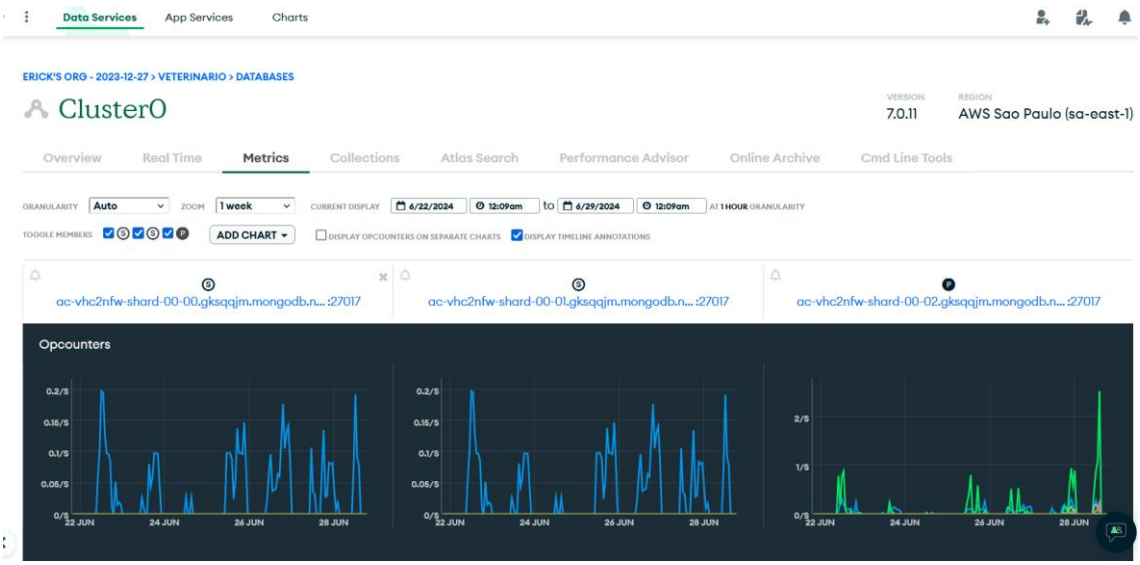


Figura 3.51: Base de datos MongoDB Atlas

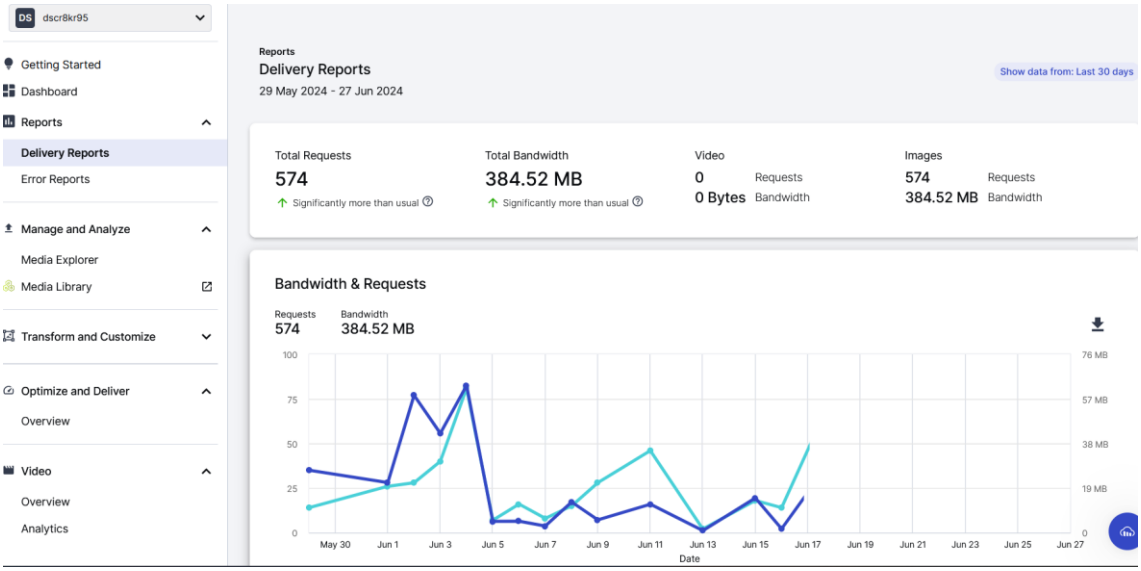


Figura 3.52: Conexión con Cloudinary

4 CONCLUSIONES

- El desarrollo correcto de una API Rest empieza al definir de forma correcta los requerimientos, estos permiten analizar de mejor manera las necesidades del usuario permitiendo definir un mejor enfoque de desarrollo de cada apartado de la API.
- El diseño de las colecciones de la base de datos permite analizar y planificar de una manera más sólida el desarrollo de la API Rest, permitiendo la planificación de Sprints mucho más eficientes.
- Implementar una arquitectura MVC para el desarrollo de la API Rest permite organizar de mejor manera todos los componentes a desarrollar, y de igual manera poder desarrollar los diferentes componentes por partes y verificar su funcionalidad.
- Al realizar diversas pruebas con el componente desarrollado se puede verificar la funcionalidad y rendimiento de la API Rest, a la vez permite identificar errores en algún componente y solucionarlos inmediatamente.
- El despliegue de la API Rest en Render ofrece ventajas simples pero útiles para cualquier aplicación, una de ellas la facilidad de despliegue y su configuración, además permite vincular el proyecto a un repositorio de GitHub facilitando el control de versiones una vez desplegado el backend.

5 RECOMENDACIONES

- Se recomienda mejorar los planes para el servicio de despliegue y las herramientas de la nube, debido que se espera un mayor número de usuarios conectados a los aplicativos web y móvil.
- Se sugiere mantener confidencialidad de las credenciales proporcionadas en este documento, para mantener la integridad y seguridad de la información existente en las aplicaciones.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] W. Martínez, «LavanGuardia,» 3 Abril 2021. [En línea]. Available: https://www.lavanguardia.com/andro4all/aplicaciones-gratis/aplicaciones-de-moda-para-descargar-gratis-en-google-play. [Último acceso: 24 Abril 2024].

[2] Anonimo, «RD STATION,» [En línea]. Available: https://www.rdstation.com/es/redes-sociales/. [Último acceso: 24 Abril 2024].

[3] «AWS.Amazon,» [En línea]. Available: https://aws.amazon.com/es/compare/the-difference-between-frontend-and-backend/. [Último acceso: 24 Abril 2024].

[4] «ayudaley,» [En línea]. Available: https://ayudaleyprotecciondatos.es/bases-de-datos/no-relacional/. [Último acceso: 24 Abril 2024].

[5] F. Antonio, «antonioFernandez,» 9 Junio 2016. [En línea]. Available: https://antoniofernandez.com/optimizar-carga-de-imagenes-cloudinary/. [Último acceso: 24 Abril 2024].

[6] F. L. Jesus, «openwebinars,» 4 Septiembre 2019. [En línea]. Available: https://openwebinars.net/blog/que-es-nodejs/. [Último acceso: 24 Abril 2024].

[7] S. R. Isabel, «Psicologia y Mente,» 8 Marzo 2018. [En línea]. Available: https://psicologiaymente.com/psicologia/estudio-de-caso. [Último acceso: 24 Abril 2024].

[8] D. Claire, «ATLASSIAN,» [En línea]. Available: https://www.atlassian.com/es/agile/scrum. [Último acceso: 24 Abril 2024].

[9] D. D. M. Ángel, «We are marketing,» 9 Mayo 2022. [En línea]. Available: https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html. [Último acceso: 24 Abril 2024].

[10] «DbaExperts.Tech,» 27 Diciembre 2022. [En línea]. Available: https://dbaexperts.tech/wp/database/levantamiento-de-requerimientos/. [Último acceso: 24 Abril 2024].

[11] «Miro,» [En línea]. Available: https://miro.com/es/agile/que-son-artefactos-scrum/. [Último acceso: 24 Abril 2024].

[12] P. Huet, «OpenWebinars,» 24 Agosto 2022. [En línea]. Available: https://openwebinars.net/blog/arquitectura-de-software-que-es-y-que-tipos-existen/. [Último acceso: 24 Abril 2024].

[13] «IBM,» [En línea]. Available: https://www.ibm.com/es-es/topics/data-architecture. [Último acceso: 24 Abril 2024].

[14] Á. A. Miguel, «Desarrollo Web,» 20 Septiembre 2023. [En línea]. Available: https://desarrolloweb.com/articulos/que-es-mvc.html. [Último acceso: 24 Abril 2024].

[15] F. Frankier, «OpenWebinars,» 22 Julio 2022. [En línea]. Available: https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/. [Último acceso: 26 Abril 2024].

[16] C. Ezequiel, «freeCodeCamp,» 14 Febrero 2021. [En línea]. Available: https://www.freecodecamp.org/espanol/news/git-vs-github-what-is-version-control-and-how-does-it-work/. [Último acceso: 26 Abril 2024].

[17] «e-dea.co,» [En línea]. Available: https://www.e-dea.co/mongodb-atlas-base-de-datos-multicloud. [Último acceso: 26 Abril 2024].

[18] «4Geeks,» [En línea]. Available: https://4geeks.com/es/lesson/despliega-modelos-ai-en-render-com-usando-flask. [Último acceso: 26 Abril 2024].

[19] P. Federico, «Latirus,» 23 Abril 2021. [En línea]. Available: https://www.latirus.com/blog/2021/04/23/thunder-client-la-alternativa-a-postman-en-vscode/. [Último acceso: 26 Abril 2024].

[20] «juntadeandalucia,» [En línea]. Available: https://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/388. [Último acceso: 26 Abril 2024].

[21] «Express,» [En línea]. Available: https://expressjs.com/es/. [Último acceso: 27 Abril 2024].

[22] L. M. L. Miguel, «OpenWebinars,» 17 Enero 2020. [En línea]. Available: https://openwebinars.net/blog/que-es-json-web-token-y-como-funciona/. [Último acceso: 26 Abril 2024].

[23] «izertis,» 9 Marzo 2017. [En línea]. Available: https://www.izertis.com/es/-/blog/encriptacion-de-password-en-nodejs-y-mongodb-bcrypt. [Último acceso: 26 Abril 2024].

[24] «YARN,» [En línea]. Available: https://classic.yarnpkg.com/en/package/fs-extra. [Último acceso: 26 Abril 2024].

[25] «Blog.Jortilles,» 11 Marzo 2024. [En línea]. Available: https://blog.jortilles.com/momentjs-la-solucion-a-tus-problemas-de-fecha/. [Último acceso: 26 Abril 2024].

[26] Diego, «Medium,» 11 Julio 2023. [En línea]. Available: https://medium.com/@diego.coder/subida-de-archivos-con-node-js-express-y-multer-55e99219d754. [Último acceso: 26 Abril 2024].

[27] «freeCodeCamp,» 16 Agosto 2023. [En línea]. Available: https://www.freecodecamp.org/espanol/news/como-usar-nodemailer-para-enviar-correos-electronicos-desde-tu-servidor-node-js/. [Último acceso: 26 Abril 2024].

[28] «Cloudinary,» [En línea]. Available: https://cloudinary.com/documentation/node_integration. [Último acceso: 26 Abril 2024].

[29] A. Torres, «freeCodeCamp,» 15 Marzo 2023. [En línea]. Available: https://www.freecodecamp.org/espanol/news/introduccion-a-mongoose-para-mongodb/. [Último acceso: 26 Abril 2024].

[30] «OpenText,» [En línea]. Available: https://www.opentext.com/es-es/que-es/functional-testing. [Último acceso: 27 Junio 2024].

[31] «OpenText,» [En línea]. Available: https://www.opentext.com/es-es/que-es/load-testing. [Último acceso: 27 Junio 2024].

[32] «OpenText,» [En línea]. Available: https://www.opentext.com/es-es/que-es/performance-testing. [Último acceso: 27 Junio 2024].

7 ANEXOS

Los anexos imparten la información necesaria para exponer de manera clara el desarrollo de este proyecto de integración curricular.

- **ANEXO I.** Certificado de Turnitin
- **ANEXO II.** Manual técnico
- **ANEXO III.** Manual de usuario (video)
- **ANEXO IV.** Manual de instalación

ANEXO I

F_AA_236

CERTIFICADO DE ORIGINALIDAD

TRABAJO DE INTEGRACIÓN CURRICULAR

Quito, D.M. 18 de julio de 2024

De mi consideración:

Yo, IVONNE FERNANDA MALDONADO SOLIZ, en calidad de Director del Trabajo de Integración Curricular titulado DESARROLLO DE UN BACKEND asociado al DESARROLLO DE UNA APLICACIÓN TIPO RED SOCIAL DE "ESTILOS DE ROPA" elaborado por el estudiante ERICK ESTIVEN RUIZ CHILUISA de la carrera en TECNOLOGÍA SUPERIOR EN DESARROLLO DE SOFTWARE, certifico que he empleado la herramienta antiplagio "TURNITIN" para la revisión de originalidad del documento escrito producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 8%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el informe generado por la herramienta Turnitin.

Atentamente,

Ivonne Maldonado
Docente Ocasional a Tiempo Completo
ESFOT

ANEXO II

Este apartado presenta la información requerida que respalda la implementación del presente proyecto.

Levantamiento de requerimientos

En la **Tabla 1**, se observan los Requerimientos necesarios para el desarrollo del backend.

Tabla 1: Levantamiento de requerimientos

Levantamiento de requerimientos		
Tipo de sistema	**ID-RR**	**ítem**
BACKEND	**RR01**	Independiente del rol para la aplicación web y móvil, se requerirá generar endpoint para: • Registrarse • Iniciar Sesión • Cambiar contraseña por primera vez (rol moderador)
	RR02	Para rol usuario, se requerirá generar endpoints para: • Visualizar perfil y publicaciones propias • Actualizar perfil • Actualizar contraseña • Recuperar contraseña • Visualizar publicaciones Favoritos
	RR03	Para rol usuario, se requerirá generar endpoints para: • Visualizar las publicaciones • Publicar fotos • Modificar la información de la publicación • Eliminar publicaciones
	RR04	Para rol de usuario, se requerirá generar endpoints para: • Reaccionar a las publicaciones • Eliminar reacción a las publicaciones • Reportar publicaciones • Agregar publicación a favoritos • Eliminar publicación de favoritos • Buscar publicaciones por medio de filtros
	RR05	Para rol de invitado, se requerirá generar endpoints para: • Visualizar publicaciones globales
	RR06	Para rol de moderador, se requerirá generar endpoints para: • Visualizar todos los usuarios • Recuperar contraseña del Moderador • Visualizar los reportes enviados • Bloquear usuarios con infracciones • Restringir la actividad del usuario • Eliminar publicaciones con infracciones • Desbloquear al usuario • Eliminar restricción al usuario • Eliminar usuarios no confirmados • Eliminar moderadores

		• Visualizar moderadores
	RR07	Para rol usuario y moderador, se requerirá generar endpoints para: • Gestionar (Crear y almacenar) notificaciones dentro de la aplicación
	RR08	Desplegar el sistema backend

Historias de usuario

Después de levantar requerimientos, se empieza la creación de las Historias de Usuarios para el backend. Se muestran las siguientes 7 historias de usuarios de los requerimientos levantados, mismos que van desde **Tabla 2** la hasta la **Tabla 7**

Tabla 2: Historia de usuario 002 – Perfil de Usuario

HISTORIA DE USUARIO	
Identificador: HU002	**Usuario:** rol usuario
Nombre historia: Funcionalidades del perfil del usuario	
Prioridad en Aplicación: Alta	**Riesgo en Aplicación:** Alta
Iteración asignada: 1	
Responsable: Erick Ruiz	
Descripción: **Como:** Usuario **Quiero:** Actualizar y visualizar mi información, así como recuperar mi contraseña **Para:** Configurar mi perfil de acuerdo a mis gustos	
Alcance: 1. El endpoint de visualizar perfil permitirá visualizar la información personal del usuario y sus publicaciones. 2. El endpoint de actualizar perfil, permitirá al usuario modificar los siguientes campos de su cuenta: a. Foto de perfil b. Descripción del usuario 3. El endpoint de actualizar contraseña, permitirá al usuario cambiar su contraseña mediante un formulario con los siguientes campos: a. Contraseña antigua b. Contraseña nueva 4. El endpoint de recuperar contraseña, permitirá recuperar contraseñas en caso de que el usuario la olvide. 5. El endpoint de visualizar Favoritos, permitirá al usuario visualizar las publicaciones favoritas que haya agregado y que aun existan dentro de la aplicación.	
Observación: La configuración o actualización de foto de perfil y descripción de usuario se realizará una vez se haya ingresado a la aplicación móvil. El proceso de recuperación de cuenta se realizará por un proceso de validación de correo electrónico.	

Criterios de Aceptación: • El usuario puede configurar su foto de perfil y su descripción una vez iniciada sesión dentro de la aplicación. • El usuario puede actualizar sus datos personales (nombre, apellido, descripción) en un solo formulario. • El usuario puede actualizar su foto de perfil. • Recuperar contraseña, este proceso se realiza por medio del envío de un código único temporal de 24 horas, dicho código deberá ser ingresado por el usuario dentro de la aplicación móvil, a su vez ingresando la nueva contraseña y confirmándola. • Las publicaciones favoritas agregadas, serán eliminadas de forma automática cuando la publicación original sea eliminada de la aplicación.

Tabla 3: Historia de usuario 003 - Publicaciones

HISTORIA DE USUARIO	
Identificador: HU003	**Usuario:** Rol usuario
Nombre historia: Funcionalidades del módulo publicaciones	
Prioridad en Aplicación: Alta	**Riesgo en Aplicación:** Alta
Iteración asignada: 2	
Responsable: Erick Ruiz	
Descripción: **Como:** Usuario **Quiero:** Interactuar con la aplicación y otros usuarios **Para:** Compartir mis publicaciones y visualizar las publicaciones de otros usuarios	
Alcance: 1. El endpoint visualizar publicaciones permitirá ver todas las publicaciones más recientes de los usuarios 2. El endpoint publicar permitirá a los usuarios crear una nueva publicación llenando los siguientes campos: a. Foto del estilo b. Descripción del estilo c. Temporada del estilo d. Año del estilo e. Genero del estilo (A que genero está dirigido el estilo) f. Estilo al que pertenece 3. El endpoint modificar la información de la publicación permite modificar los siguientes campos: a. Descripción del estilo b. Temporada del estilo c. Año del estilo d. Genero del estilo e. Estilo al que pertenece 4. El endpoint eliminar publicación permitirá al usuario eliminar la publicación que no desee mostrar en su perfil.	
Observación: • Las publicaciones solo estarán disponibles para la aplicación móvil. • **Foto de estilo (campo no editable):** La foto deberá ser cargada desde la galería de los usuarios.	

• **Descripción del estilo (campo de texto):** Se detallará lo más destacable del estilo a publicar. • **Temporada del estilo (Campo de texto):** El usuario seleccionara a que temporada del año pertenece su estilo (invierno, verano, otoño, etc.). • **Año del estilo (Campo de texto):** El usuario seleccionara a que época pertenece su estilo (70's, 80's 90's, 2000's, etc.). • **Genero del estilo:** El usuario seleccionara a que genero está dirigido su estilo (Estilo Masculino, Estilo Femenino, Estilo para todos). • **Estilo:** El usuario podrá seleccionar el estilo que corresponda a su publicación (Casual, Formal, Etiqueta, Deportivo, etc.).
Criterios de aceptación: • Las publicaciones solo se podrán publicar una vez llenos todos los campos. • El usuario podrá actualizar los detalles (Descripción, temporada del estilo, año del estilo, genero del estilo, estilo) de su publicación según lo requiera. • El usuario podrá eliminar su publicación en caso de requerirlo. • Se visualizará en un panel de publicación todas las publicaciones de los usuarios y se actualizará en tiempo real. • Las fotos serán insertadas desde los archivos del usuario

Tabla 4: Historia de usuario 004 - Interacciones

HISTORIA DE USUARIO	
Identificador: HU004	**Usuario:** Rol Usuario
Nombre historia: Interacción con otros usuarios	
Prioridad en Aplicación: Alta	**Riesgo en Aplicación:** Media
Iteración asignada: 2	
Responsable: Erick Ruiz	
Descripción: **Como:** Usuario **Quiero:** Interactuar con las publicaciones de otros usuarios **Para:** Reaccionar mediante like, dislike, agregar a favoritos o reportar.	
Alcance: 1. El endpoint Reaccionar a publicaciones permitirá al usuario agregar un like o dislike a las publicaciones de otros usuarios. 2. El endpoint Reportar publicación permitirá al usuario indicar su descontento ante una publicación que considere inadecuada mediante un formulario con los siguientes campos: a. Motivo del reporte b. Descripción del reporte 3. El endpoint Agregar a favoritos permitirá al usuario almacenar los estilos publicados de su mayor agrado. 4. El endpoint Eliminar reacción permitirá al usuario borrar su like o dislike de la publicación de otro usuario. 5. El endpoint Eliminar de favoritos permitirá al usuario eliminar las publicaciones de su lista de favoritos. 6. El endpoint Buscar permitirá al usuario filtrar las publicaciones según desee.	
Observación: • El conteo de reacciones (Like y dislike) podrá ser visualizada por todos los usuarios y moderadores. • Los usuarios que den reportes falsos no serán amonestados a menos que reincidan en cierto número (mayor a 10) de reportes falso.	

• Los usuarios serán notificados por cada acción dentro de sus publicaciones (like, dislike y reporte)
Criterios de aceptación: • Cada usuario podrá reaccionar una vez a cada publicación • El reporte enviara un formulario con los campos llenados a los moderadores, mismos que determinarán la veracidad del reporte y darán sanción al reportado. • Los moderadores darán notificación de la sanción en caso de ser confirmado el reporte. • En caso de ser reporte falso se dará una notificación indicando dicho suceso. • Los reportes no serán almacenados, serán temporales. • Solo los moderadores podrán visualizar al usuario que reporte una publicación. • Agregar a favoritos no será notificado a los usuarios dueños de dichas publicaciones. • El usuario puede buscar sus estilos por medio de filtros establecidos.

Tabla 5: Historia de usuario 005 - Rol invitado

HISTORIA DE USUARIO	
Identificador: HU005	**Usuario:** Rol Invitado
Nombre historia: Visitante de la aplicación web	
Prioridad en Aplicación: Media	**Riesgo en Aplicación:** Baja
Iteración asignada: 4	
Responsable: Erick Ruiz	
Descripción: **Como:** Invitado **Quiero:** Visualizar en la web las publicaciones de los usuarios registrados **Para:** Interesarme en las funcionalidades que ofrece la aplicación móvil.	
Alcance: 1. El endpoint de visualizar publicaciones, permitirá a los invitados de la aplicación web visualizar las interacciones de los usuarios registrados.	
Observación: • Los invitados solo podrán visualizar las publicaciones, pero no podrán interactuar con las publicaciones. • Los invitados solo podrán ingresar desde la aplicación web.	
Criterios de aceptación: • La aplicación web permitirá el ingreso de invitados, pero no tendrá módulo de registro o inicio de sesión para usuarios.	

Tabla 6: Historia de usuario 006 - Funciones del moderador

HISTORIA DE USUARIO	
Identificador: HU006	**Usuario:** Rol moderador
Nombre historia: Funciones del moderador	
Prioridad en Aplicación: Alta	**Riesgo en Aplicación:** Alta
Iteración asignada: 3	

Responsable: Erick Ruiz

Descripción:
Como: Moderador
Quiero: Controlar las publicaciones y usuarios dentro de la aplicación móvil
Para: Asegurar un ambiente sano y respetuoso dentro de la aplicación.

Alcance:

1. El endpoint visualizar usuarios, permite al moderador ver cada Usuario registrado y si este posee alguna sanción en su cuenta.
2. El endpoint recuperar contraseña, permitirá al moderador restablecer su contraseña en caso de haberla olvidado.
3. El endpoint visualizar reportes, permite al moderador ver los reportes enviados y que no han sido solucionados.
4. El endpoint bloquera usuario, permitirá al moderador sancionar a aquellas cuentas con contenido inadecuado dentro de la aplicación móvil.
5. El endpoint borrar publicaciones, permitirá al moderador borrar aquellas publicaciones con contenido inadecuado.
6. El endpoint falso reporte, permitirá al moderador resolver un reporte sin sancionar al usuario debido a que el reporte fue falso
7. El endpoint cambio de estado permitirá al moderador cambiar el estado a resuelto de los reportes de una publicación que ya ha sido borrada.
8. El enpoint Restringir, permite al moderador restringir temporalmente al usuario que cometió una infracción.
9. Los endpoints para desbloquear y habilitar usuarios permiten al moderador retirar las sanciones a los usuarios
10. El endpoint Borrar usuario, permitirá al moderador eliminar las cuentas de los usuarios que tengan confirmado su correo electrónico.
11. El endpoint Eliminar Moderadores, permitirá al Moderador Super eliminar a diversos moderadores de la aplicación web.
12. El endpoint visualizar moderadores confirmado, permite al moderador Super visualizar a los moderadores que confirmaron su cuenta, registrados
13. El endpoint visualizar moderadores no confirmados, permite al moderador Super visualizar a los moderadores registrados que aún no han confirmado su correo electrónico.

Observación:

- El moderador podrá realizar sus respectivas acciones desde la aplicación web.
- El moderador Super podrá realizar acciones extras, como registrar o eliminar otros moderadores.

Criterios de aceptación:

- Los reportes serán visualizados por los moderadores
- El moderador bloqueara temporal o indefinidamente las cuentas con infracciones.
- Restringir la cuenta del usuario, evitara que este pueda interactuar con la aplicación, es decir, no puede publicar ni reaccionar a otras publicaciones, por un cierto tiempo, 3, 7 y 15 días dependiendo las veces que vaya infringiendo los términos y condiciones.
- Los usuarios con 3 infracciones serán bloqueados en caso de reincidir en conducta inadecuada dentro de la aplicación.
- Los usuarios bloqueados no podrán registrarse con el correo de la cuenta bloqueada.
- Las publicaciones reportadas serán borradas una vez que el moderador verifico la veracidad del reporte.
- En caso de ser bloqueada la cuenta de forma indefinida, se notificará al usuario mediate correo electrónico.

• Se mostrará la fecha cuando se creó el usuario con la finalidad de analizar si el usuario confirmará su correo electrónico, si la cuenta no es confirmada en cierto tiempo los moderadores podrán borrarla de la base de datos. • El moderador podrá restablecer su contraseña mediante el envío de un correo electrónico que verificará al usuario y permitirá cambiar a una nueva contraseña. • Las restricciones serán retiradas por los moderadores al cumplirse el tiempo de restricción • Las cuentas bloqueadas podrán comunicarse con los moderadores por correo electrónico para recuperar su cuenta.

Tabla 7: Historia de usuario 007 - Notificaciones

HISTORIA DE USUARIO	
Identificador: HU007	**Usuario:** Rol usuario y moderador
Nombre historia: Notificaciones	
Prioridad en Aplicación: Media	**Riesgo en Aplicación:** Baja
Iteración asignada: 4	
Responsable: Erick Ruiz	
Descripción: **Como:** Usuario y moderador **Quiero:** Ver las últimas interacciones dentro de las aplicaciones **Para:** Conocer y mantenerme informado a los diversos sucesos en mi cuenta usuario o moderador	
Alcance: 1. El endpoint gestionar Notificaciones, permitirá crear y almacenar las notificaciones de cada acción dentro de las aplicaciones para cada rol: a. Rol usuario: i. Notificación de las reacciones de mis publicaciones ii. Notificación del tiempo de restricción b. Rol moderador: i. Notificación de los Últimos reportes realizados	
Observación: Los moderadores serán notificados tanto en la aplicación web como en sus correos electrónicos.	
Criterios de aceptación: • Las notificaciones serán visualizadas por cada rol en sus respectivas aplicaciones. • Las notificaciones de los moderadores serán eliminadas una vez se resuelva dicho reporte. • En las notificaciones de los usuarios serán divididas en dos tipos, de alerta y de reacción. • Las notificaciones de alerta de los usuarios mostraran el tiempo de restricción que tenga la cuenta. • Las notificaciones de reacción de los usuarios mostraran quien dio like a cierta publicación personal.	

Product Backlog

La **Tabla 8** enlista los requisitos mediante las historias de usuario según las necesidades del Product Owner, indicando su prioridad.

Tabla 8: Product Backlog

ELABORACIÓN DEL *PRODUCT BACKLOG*				
ID – HU	**HISTORIA DE USUARIO**	**ITERACIÓN**	**ESTADO**	**PRIORIDAD**
HU001	Generar un endpoints para el registrar e iniciar sesión para cualquier rol	1	Finalizado	Alta
HU002	Generar varios endpoints para las funcionalidades del perfil de usuario	1	Finalizado	Alta
HU003	Generar varios endpoints para las funcionalidades del módulo de publicaciones	2	Finalizado	Alta
HU004	Generar varios endpoints para las funcionalidades con del módulo de interacciones.	2	Finalizado	Alta
HU005	Generar varios endpoints para el ingreso de los visitantes	4	Finalizado	Media
HU006	Generar varios endpoints para las funcionalidades del módulo de moderadores.	3	Finalizado	Alta
HU007	Generar varios endpoints para las funcionalidades de las notificaciones	4	Finalizado	Media

Sprint Backlog

En la **Tabla 9** detalla los cinco Sprints para desarrollar el componente backend, en la misma tabla se muestra las tareas y el tiempo estimado de desarrollo, cada Sprint está vinculado a cierto número de Historias de Usuarios, con la finalidad de alcanzar los objetivos establecidos por el Product Owner para los entregables necesarios.

Tabla 9: Sprint Backlog

ID-SB	NOMBRE	HISTORIA DE USUARIO	TAREAS	TIEMPO ESTIMADO
S0	Configuración del ambiente del desarrollo	N/A	• Definición de roles. • Recopilación de requerimientos • Configuración de herramientas necesarias para el desarrollo. • Diseño de la Base de Datos en MongoDB. • Configuración de Cloudinary	20 h
S1	Autenticación, registro y perfil	HUB001: Registrar e iniciar sesión con cualquier rol. HUB002: Funcionalidad del perfil de usuario	• Crear endpoint para el registro de usuario y moderador. • Crear endpoint para el inicio de sesión para usuario y moderador. • Crear endpoints para las funcionalidades del perfil de usuario	50 h
S2	Módulo de publicaciones e interacciones	HUB003: Funcionalidad del módulo de publicaciones	• Crear endpoints para las funcionalidades del módulo de publicaciones para el usuario. ○ Conectividad con Cludinary	50 h

		HUB004: Funcionalidad del módulo de interacciones	• Crear endpoints para las funcionalidades del módulo de interacciones.	
S3	Módulo de moderadores	HUB006 Funcionalidad del módulo moderadores	• Crear endpoints para todas las funcionalidades del módulo de moderadores.	30h
S4	Módulo Invitados y notificaciones	HUB005: Funcionalidades para el módulo invitado HUB007: Funcionalidades para las notificaciones.	• Crear endpoints para el módulo de invitado para la aplicación web • Crear endpoints para las funcionalidades del módulo notificaciones del usuario • Crear endpoints para funcionalidades del módulo notificaciones del moderador	40 h
S5	Pruebas y Despliegue	N/A	• Pruebas de Funcionalidad. • Prueba de Carga. • Prueba de Rendimiento. • Despliegue API Rest en Render	30 h
Documentación		N/A	• Trabajo de Integración Curricular. • Anexos.	20 h
			Total =	240 h

Pruebas

En este apartado se detalla las pruebas realizadas al componente ROPDAT al finalizar su desarrollo, enfocándose en probar su funcionalidad y rendimiento.

Pruebas de Funcionalidad

Se presenta las pruebas de funcionalidad y sus resultados, desde la **Figura 1** hasta la **Figura 84**, indicando que el componente funciona a la perfección.

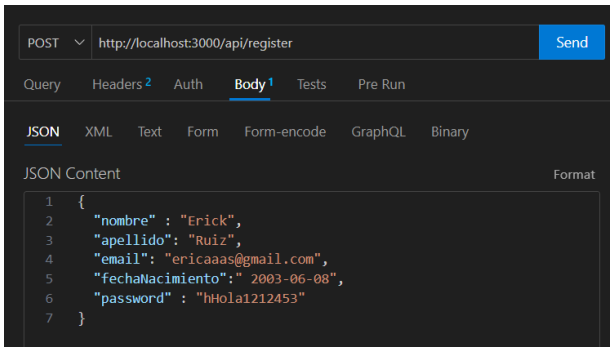


Figura 1: Prueba de funcionalidad Registro Usuario

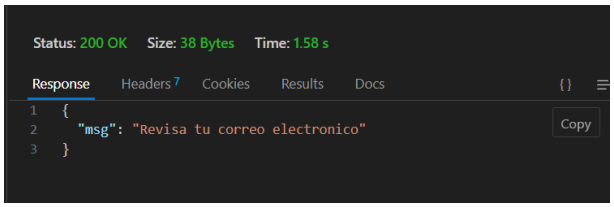


Figura 2: Resultado Prueba de Funcionalidad

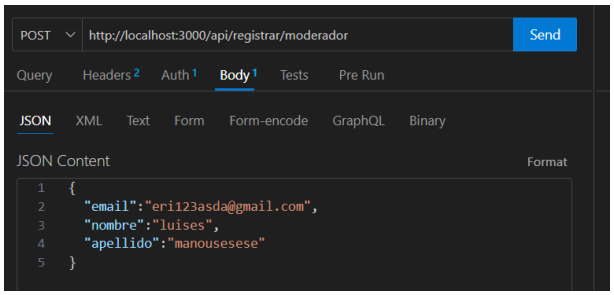


Figura 3: Prueba de Funcionalidad Registro Moderador

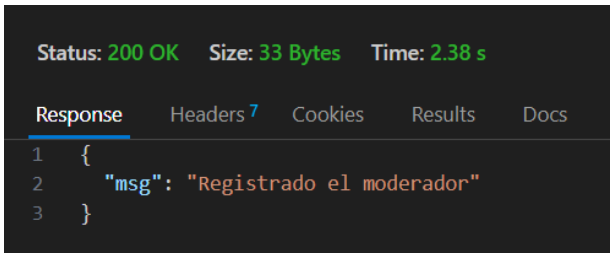


Figura 4: Resultado Prueba de Funcionalidad

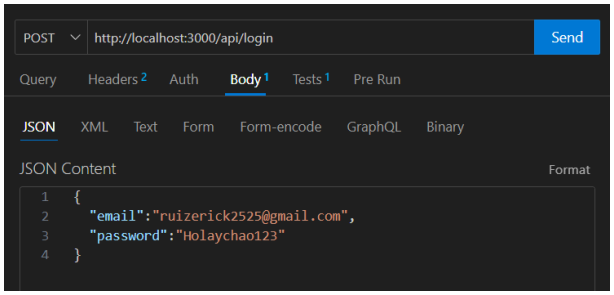


Figura 5: Prueba Funcionalidad Inicio de sesión Usuario

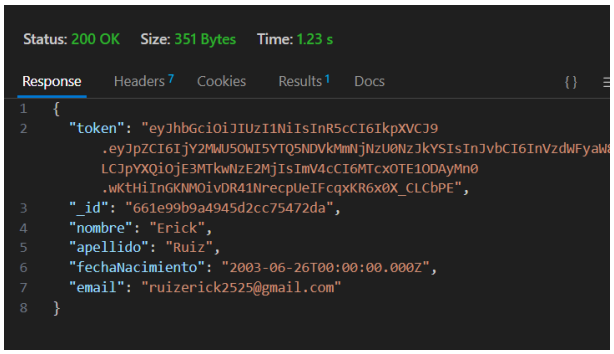


Figura 6: Resultado Prueba de Funcionalidad

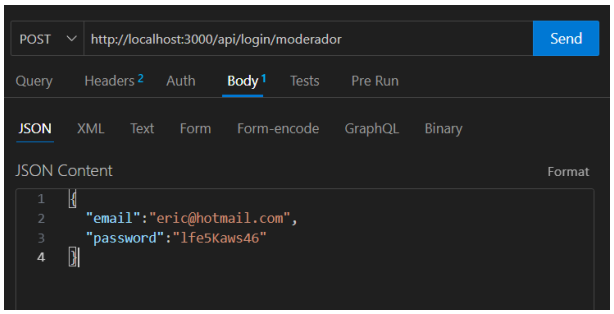


Figura 7: Prueba de Funcionalidad Inicio de sesión de Moderador

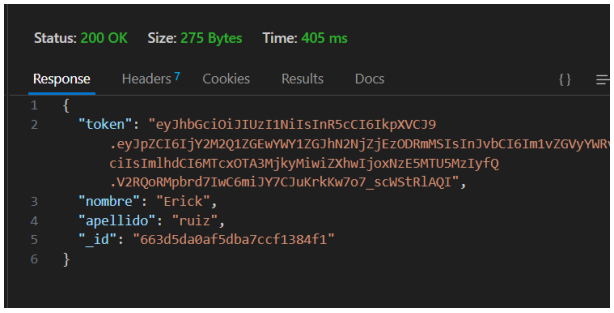


Figura 8: Resultado Prueba de Funcionalidad

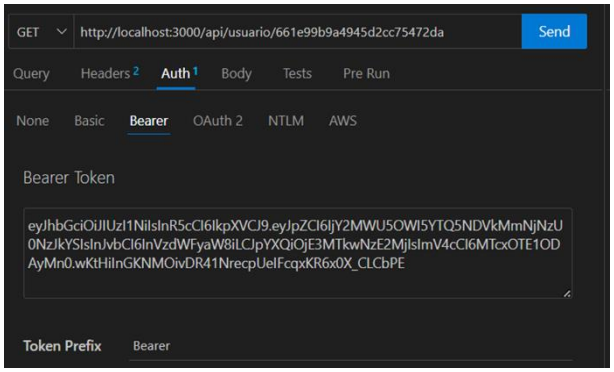


Figura 9: Prueba de Funcionalidad Perfil Usuario

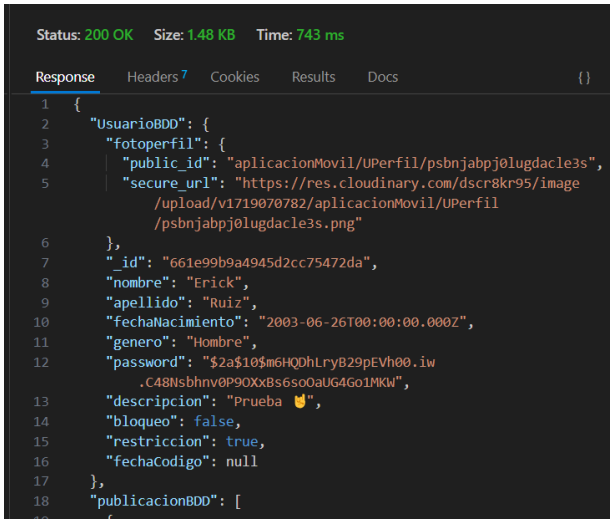


Figura 10: Resultado Prueba de Funcionalidad

Figura 11: Prueba de Funcionalidad Actualizar Perfil

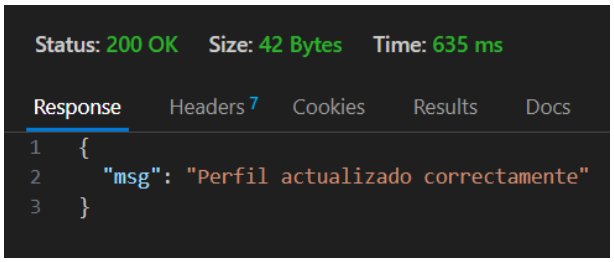


Figura 12: Resultado Prueba de Funcionalidad

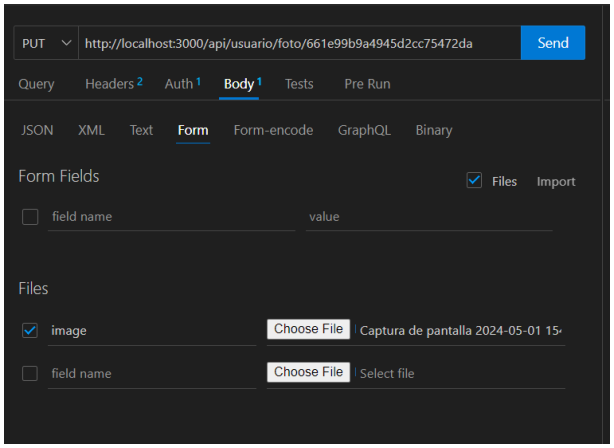


Figura 13: Prueba de Funcionalidad Actualizar Foto Usuario

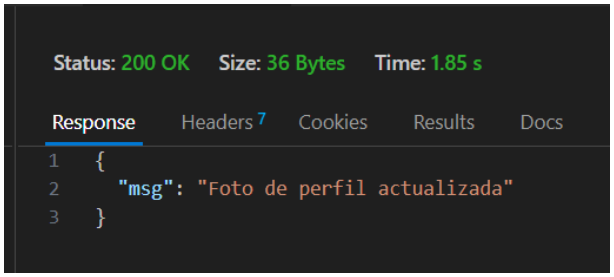


Figura 14: Resultado Prueba de Funcionalidad

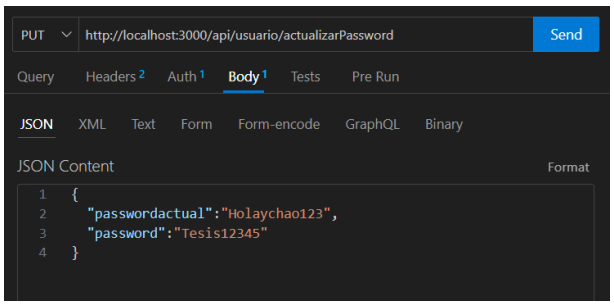


Figura 15: Prueba de Funcionalidad Actualizar Contraseña

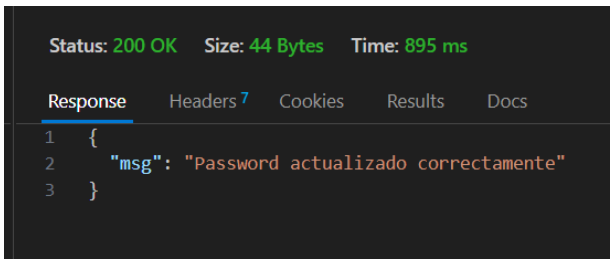


Figura 16: Resultado Prueba de Funcionalidad

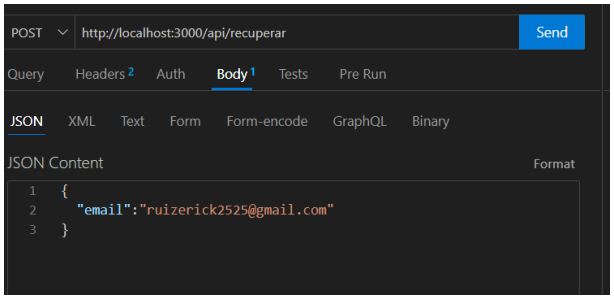


Figura 17: Prueba de Funcionalidad Restablecer contraseña

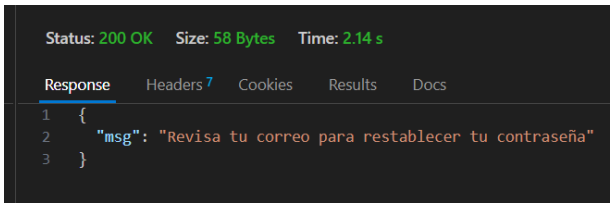


Figura 18: Resultado Prueba de Funcionalidad

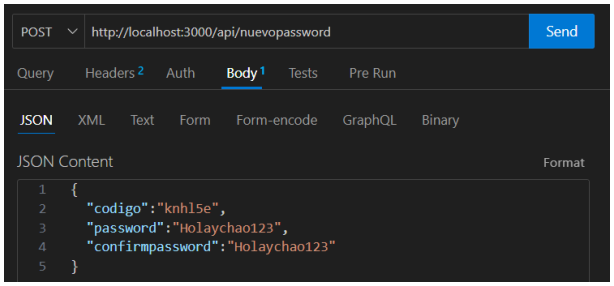


Figura 19: Prueba de Funcionalidad Recuperar Contraseña

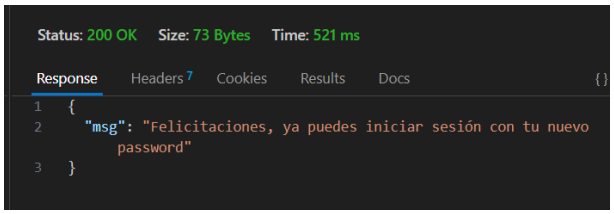


Figura 20: Resultado Prueba de Funcionalidad

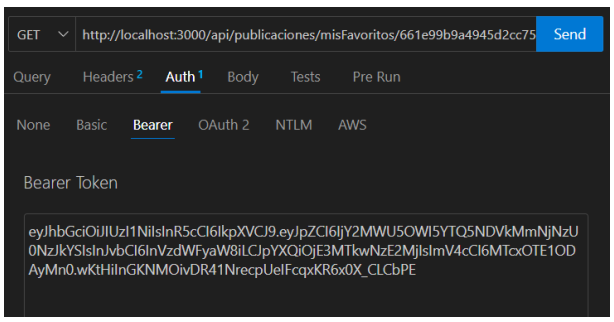


Figura 21: Prueba de Funcionalidad Visualizar Favoritos

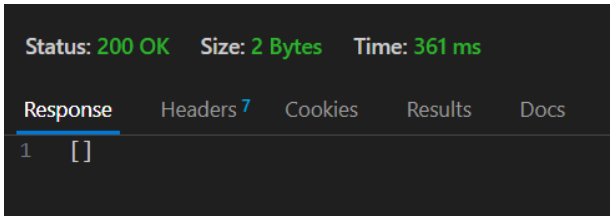


Figura 22: Resultado Prueba de Funcionalidad

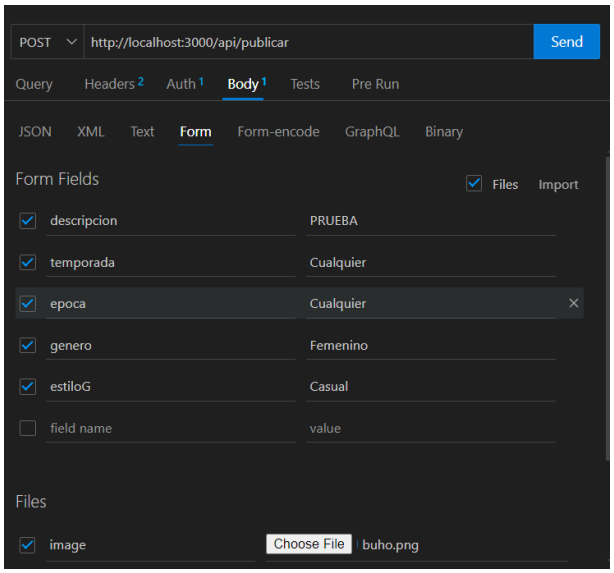


Figura 23: Prueba de Funcionalidad Publicar

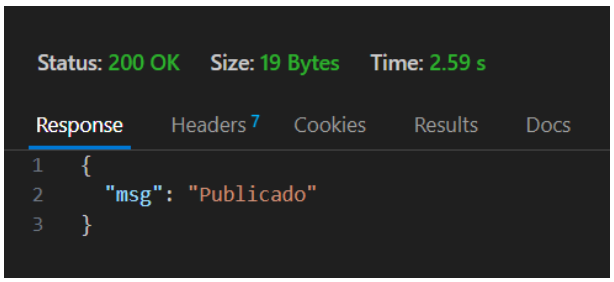


Figura 24: Resultado Prueba de Funcionalidad

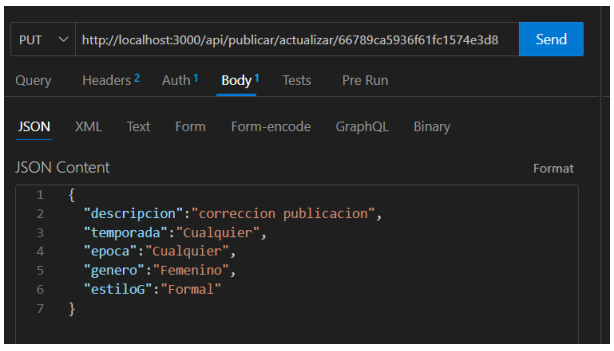


Figura 25: Prueba de Funcionalidad Actualizar Publicación

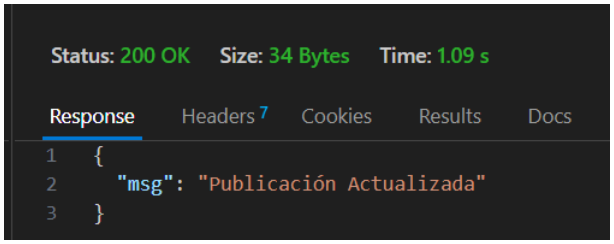


Figura 26: Resultado Prueba de Funcionalidad

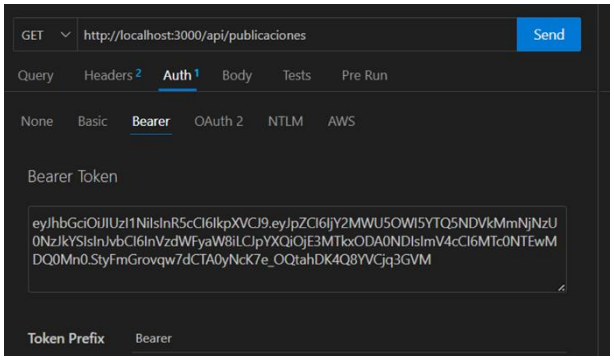


Figura 27: Prueba de Funcionalidad Ver Publicaciones

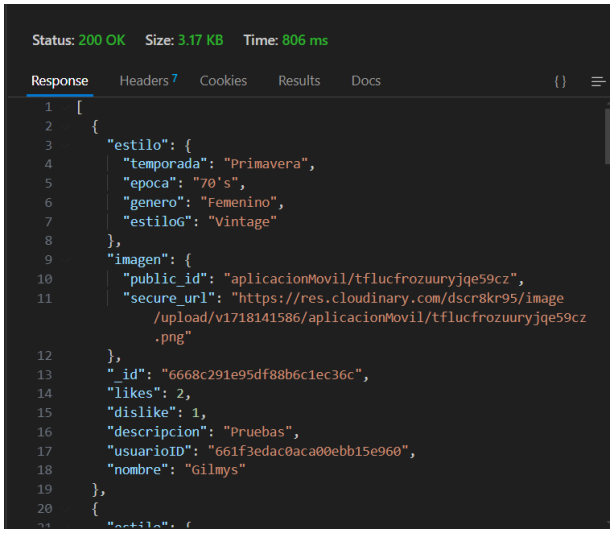


Figura 28: Resultado Prueba de Funcionalidad

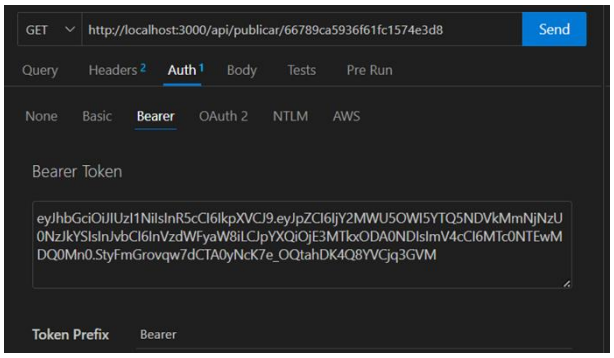


Figura 29: Prueba de Funcionalidad Ver Publicación única

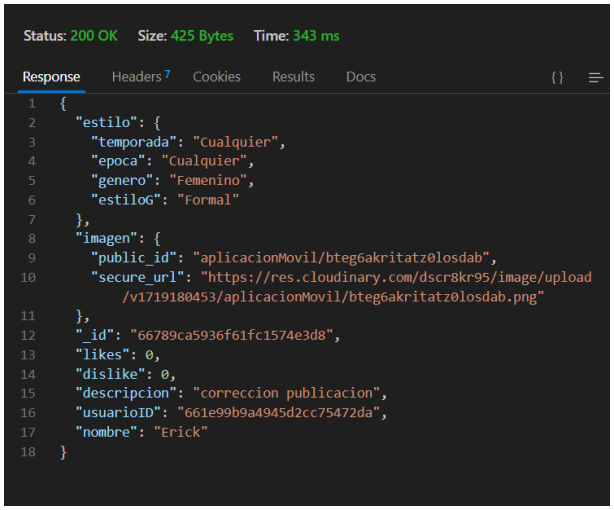


Figura 30: Resultado Prueba de Funcionalidad

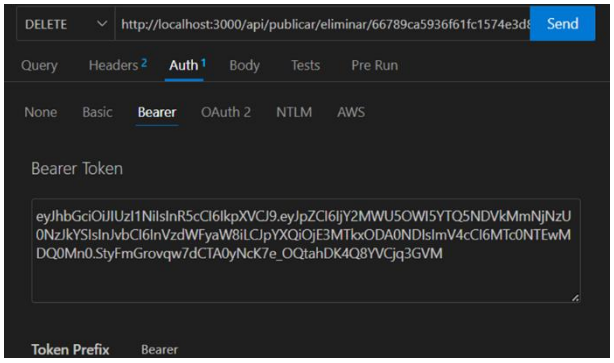


Figura 31: Prueba de Funcionalidad Eliminar Publicación

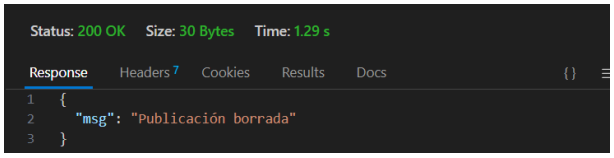


Figura 32: Resultado Prueba de Funcionalidad

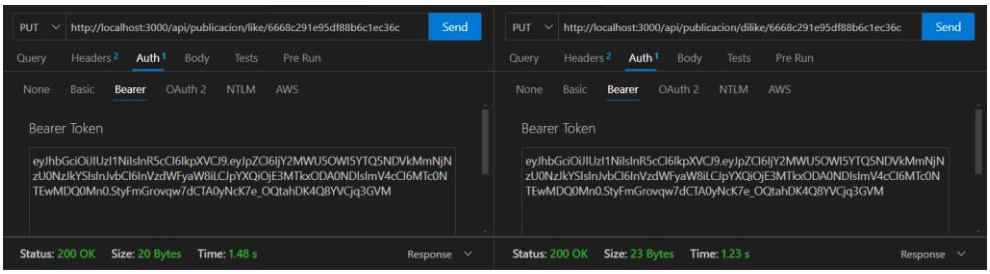


Figura 33: Prueba de Funcionalidad Dar like y Dislike

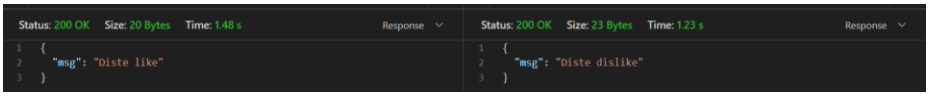


Figura 34: Resultado Prueba de Funcionalidad

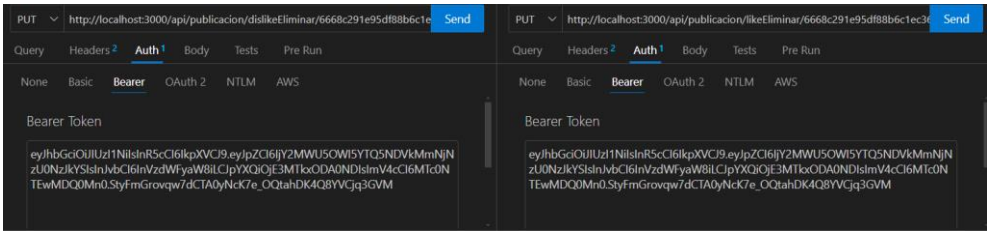


Figura 35: Prueba de Funcionalidad Eliminar Like y Dislike

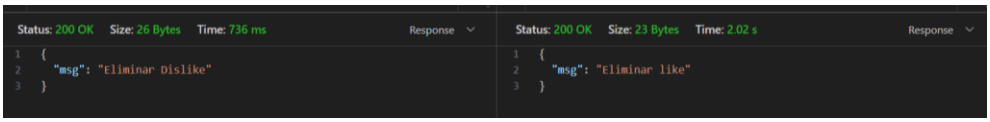


Figura 36: Resultado Prueba de Funcionalidad

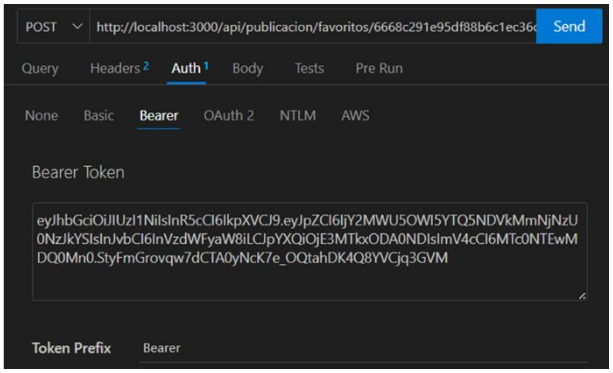


Figura 37: Pruebas de Funcionalidad Agregar a Favoritos

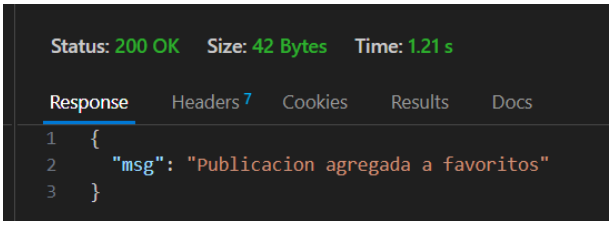


Figura 38: Resultados Prueba de Funcionalidad

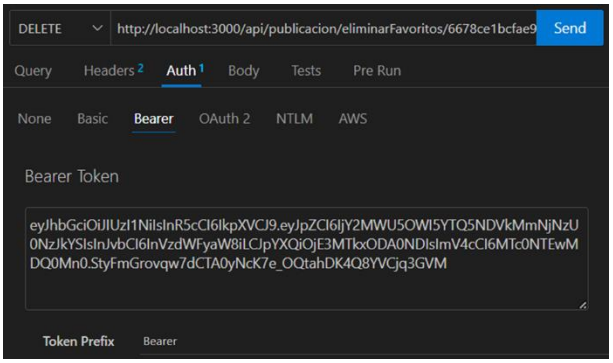


Figura 39: Prueba de Funcionalidad Eliminar Favoritos

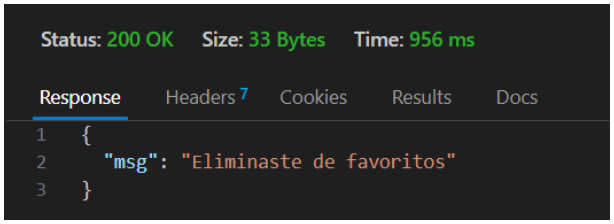


Figura 40: Resultado Prueba de Funcionalidad

Figura 41: Prueba de Funcionalidad Búsqueda

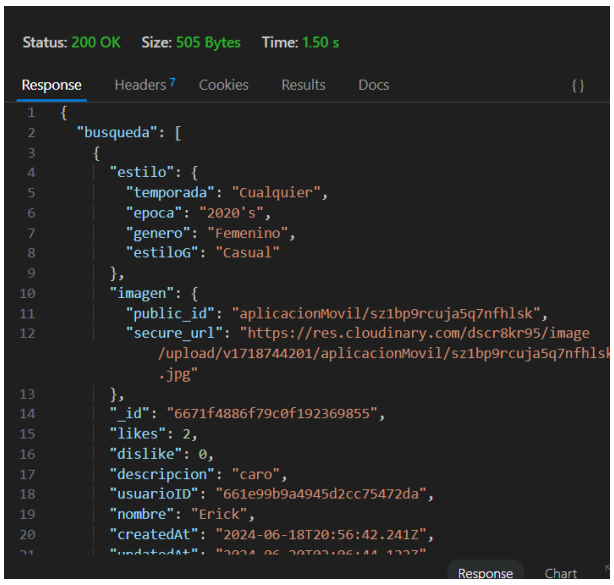


Figura 42: Resultado Prueba de Funcionalidad

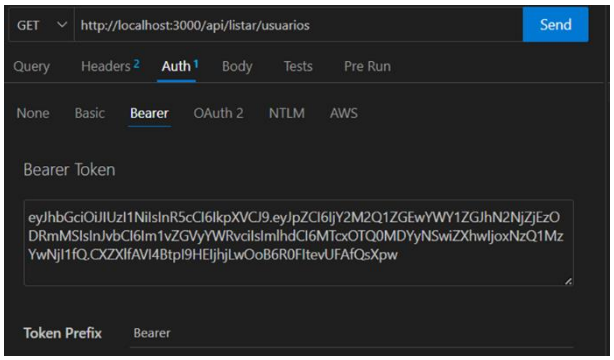


Figura 43: Prueba de Funcionalidad Listar Usuarios

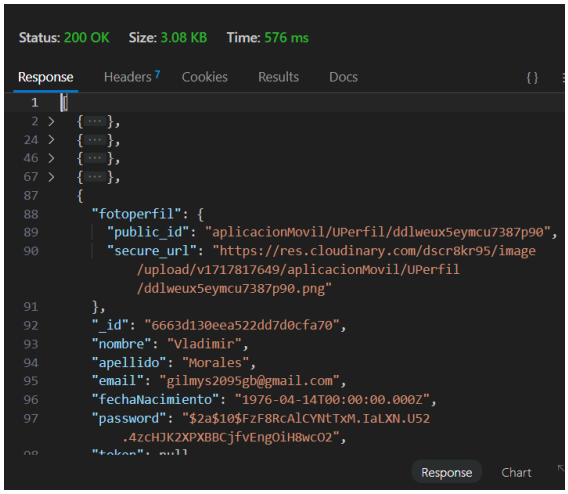


Figura 44: Resultado Prueba de Funcionalidad

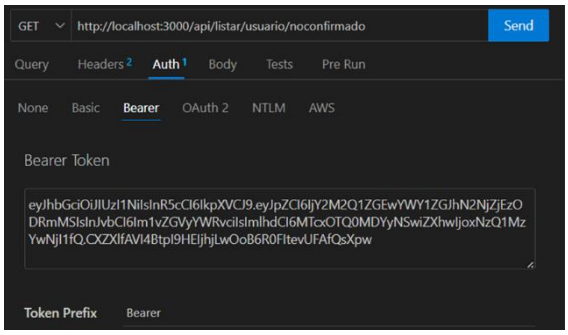


Figura 45: Prueba de Funcionalidad Listar Usuarios No Confirmados

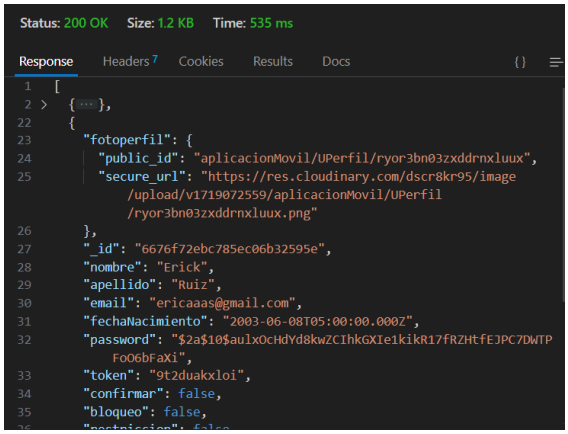


Figura 46: Resultado Prueba de Funcionalidad

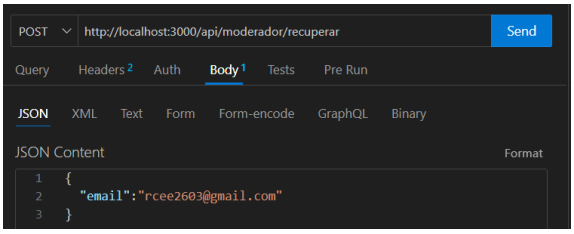


Figura 47: Prueba de Funcionalidad Enviar Correo Restablecer Contraseña

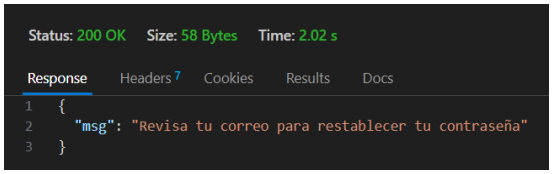


Figura 48: Resultado Prueba de Funcionalidad

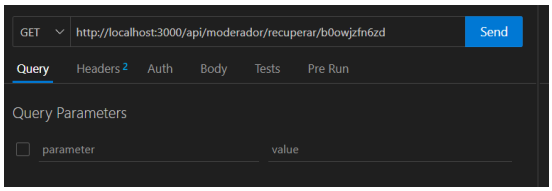


Figura 49: Prueba de Funcionalidad Confirmar Token

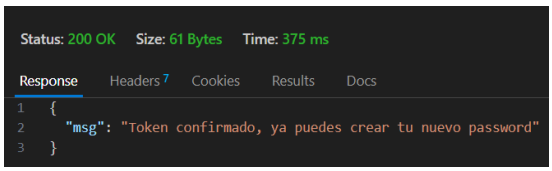


Figura 50: Resultado Prueba de Funcionalidad

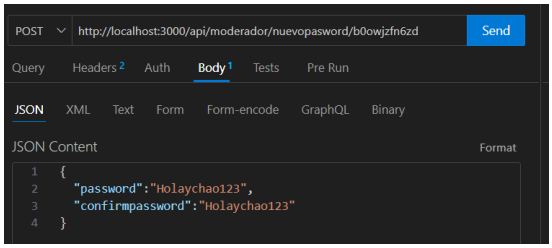


Figura 51: Prueba de Funcionalidad Restablecer Contraseña

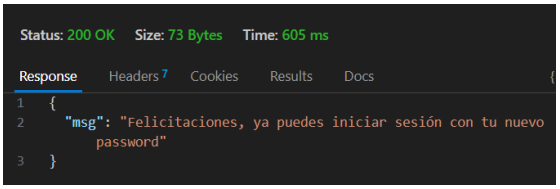


Figura 52: Resultado Prueba de Funcionalidad

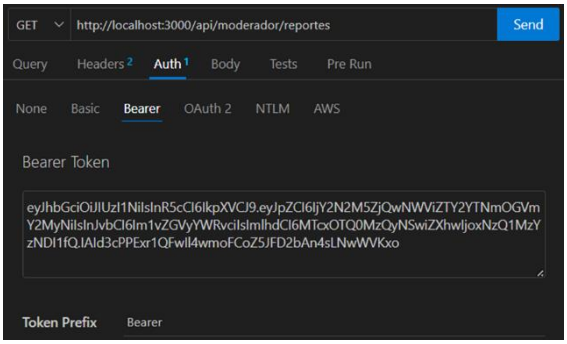


Figura 53: Prueba de Funcionalidad Listar Reportes

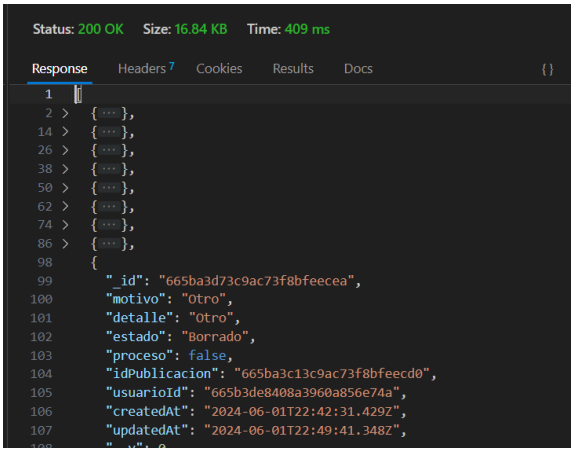


Figura 54: Resultado Prueba de Funcionalidad

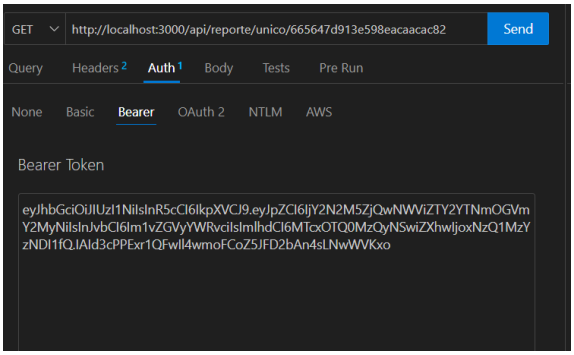


Figura 55: Prueba de Funcionalidad Visualizar Reporte

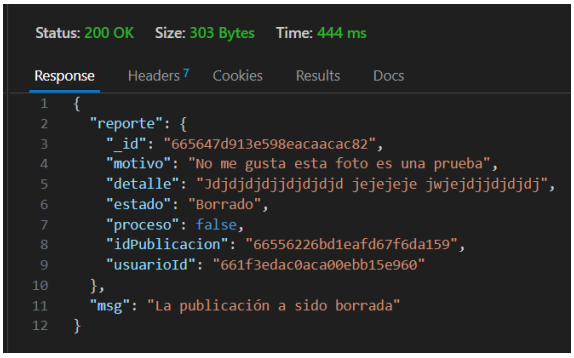


Figura 56: Resultado Prueba de Funcionalidad

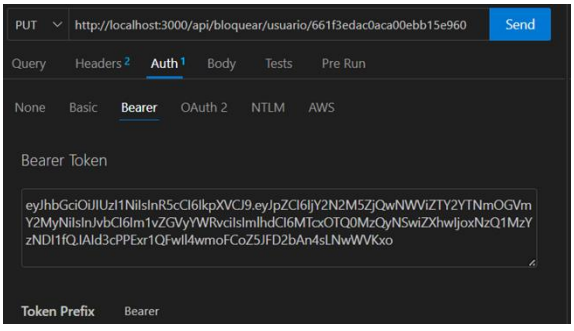


Figura 57: Prueba de Funcionalidad Bloquear Usuario

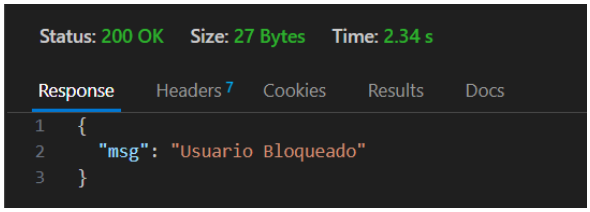


Figura 58: Resultado Prueba de Funcionalidad

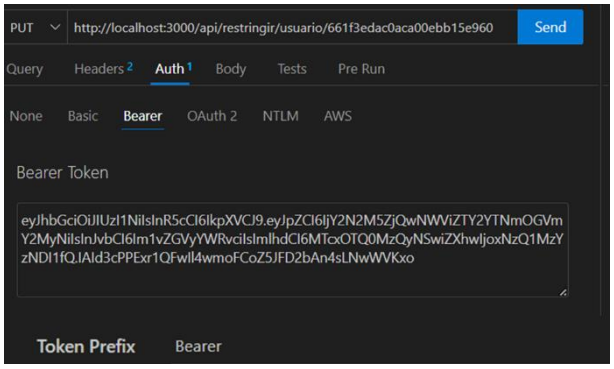


Figura 59: Prueba de Funcionalidad Restringir Usuario

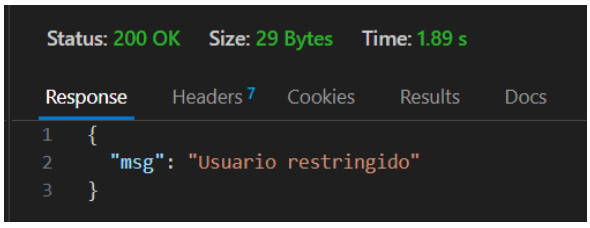


Figura 60: Resultado Prueba de Funcionalidad

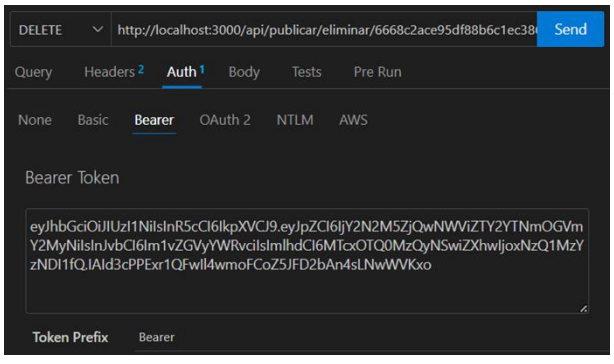


Figura 61: Prueba de Funcionalidad Borrar Publicación Reportada

Figura 62: Resultado Prueba de Funcionalidad

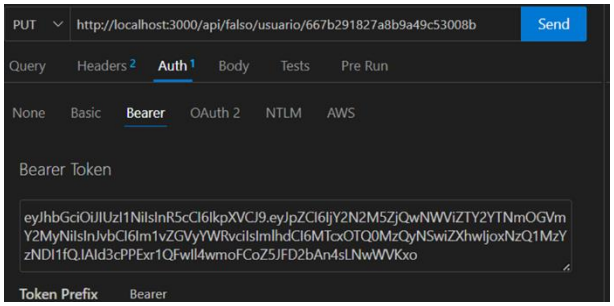


Figura 63: Prueba de Funcionalidad Reporte Falso

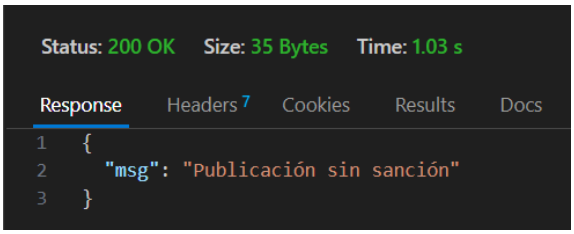


Figura 64: Resultado Prueba de Funcionalidad

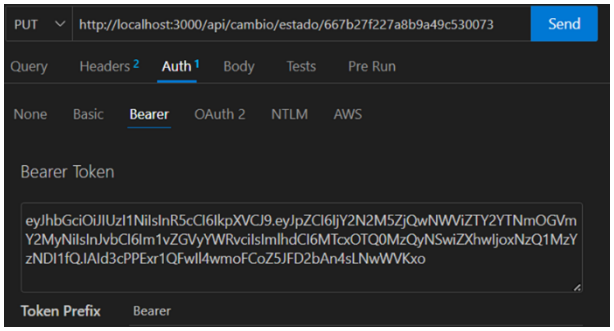


Figura 65: Prueba de Funcionalidad Cambiar Estado Reporte

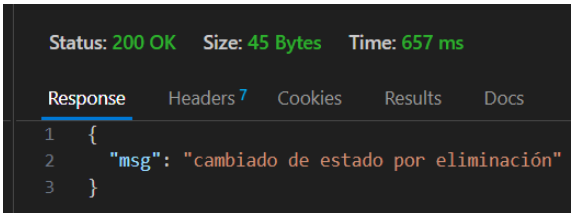


Figura 66: Resultado Prueba de Funcionalidad

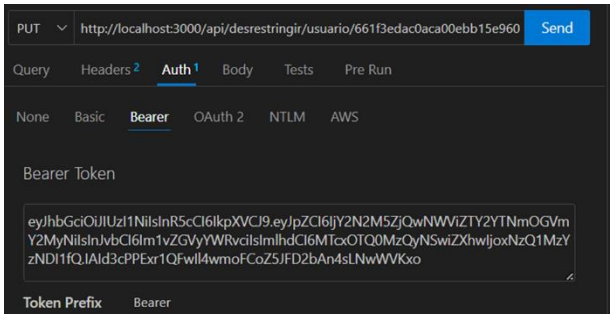


Figura 67: Prueba de Funcionalidad Habilitar Usuario

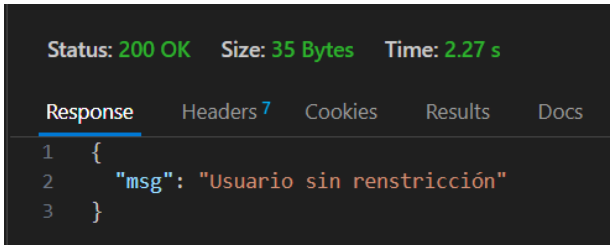


Figura 68: Resultado Prueba de Funcionalidad

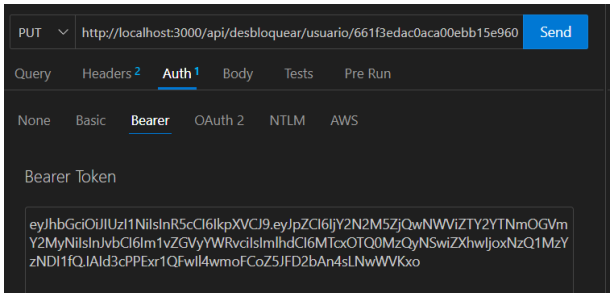


Figura 69: Prueba de Funcionalidad Desbloquear Usuario

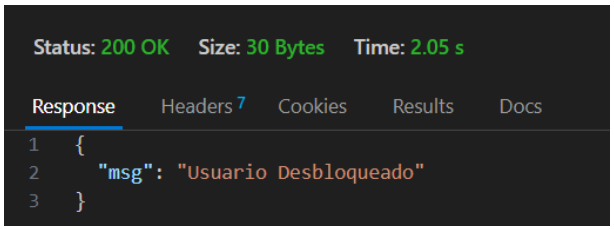


Figura 70: Resultado Prueba de Funcionalidad

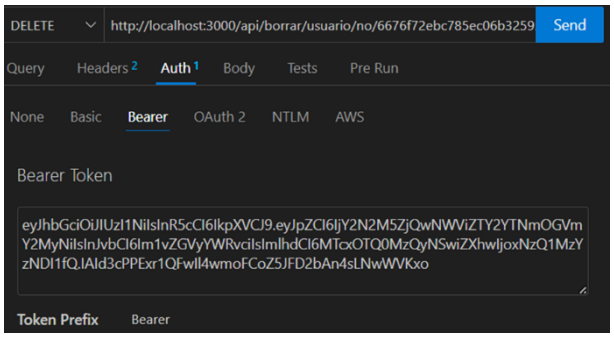


Figura 71: Prueba de Funcionalidad Borrar Usuario

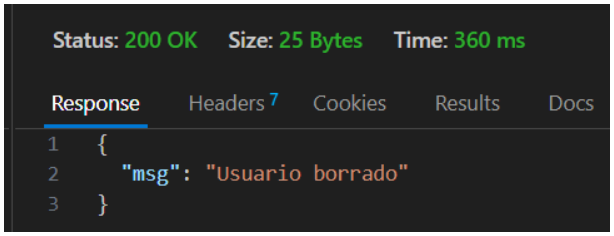


Figura 72: Resultado Prueba de Funcionalidad

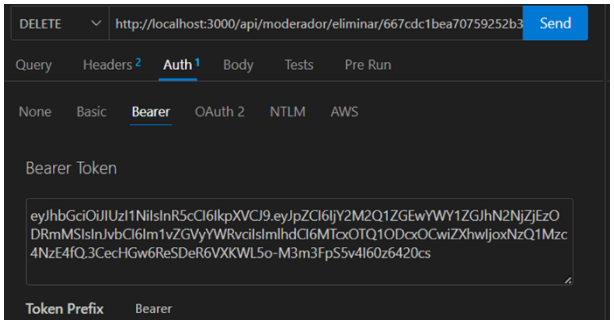


Figura 73: Prueba de Funcionalidad Eliminar Moderador

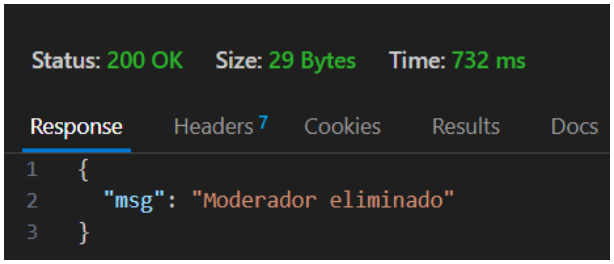


Figura 74: Resultado Prueba de Funcionalidad

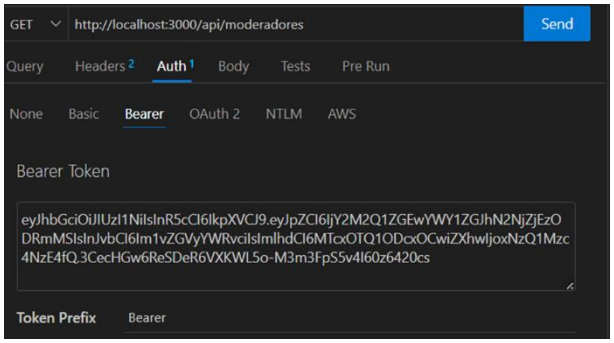


Figura 75: Prueba de Funcionalidad Listar Moderadores

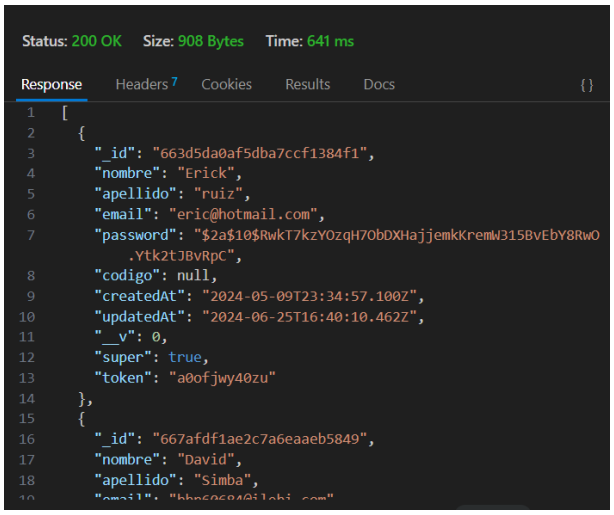


Figura 76: Resultado Prueba de Funcionalidad

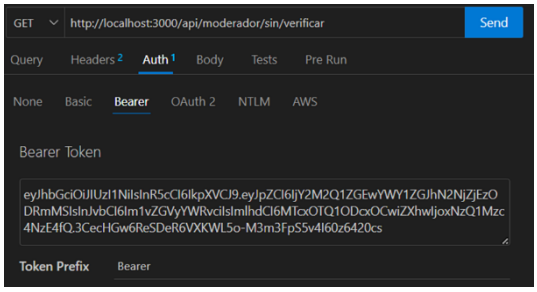


Figura 77: Prueba de Funcionalidad Listar Moderadores sin Verificar

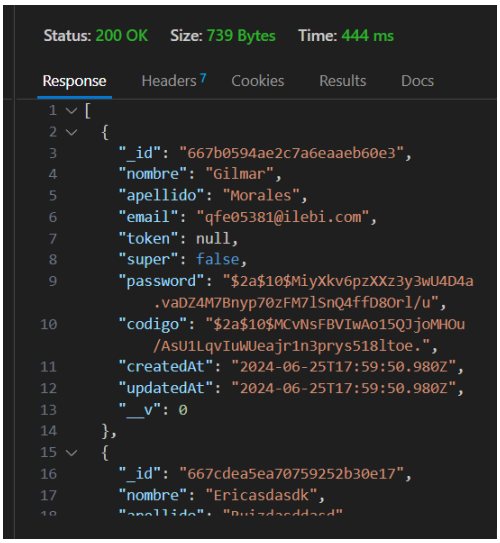


Figura 78: Resultado Prueba de Funcionalidad

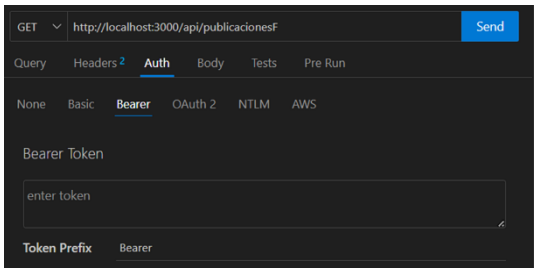


Figura 79: Prueba de Funcionalidad Ver Publicaciones Invitado

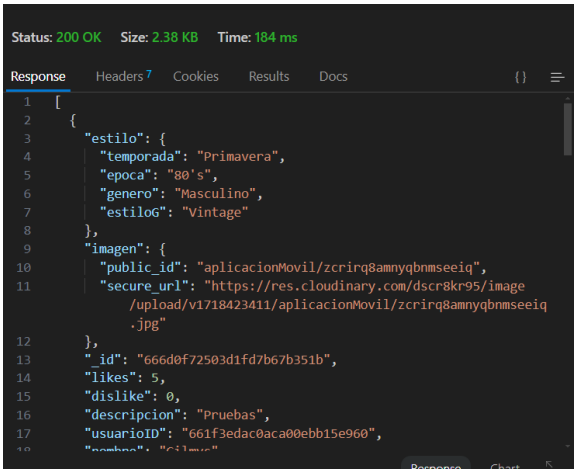


Figura 80: Resultado Prueba de Funcionalidad

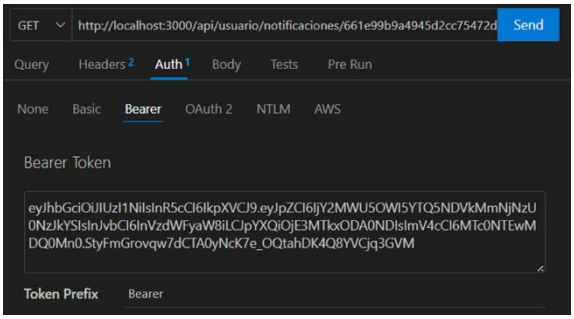


Figura 81: Prueba de Funcionalidad Listar Notificaciones Usuarios

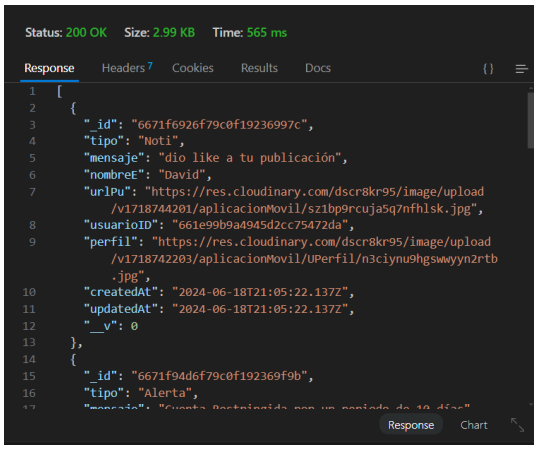


Figura 82: Resultado Prueba de Funcionalidad

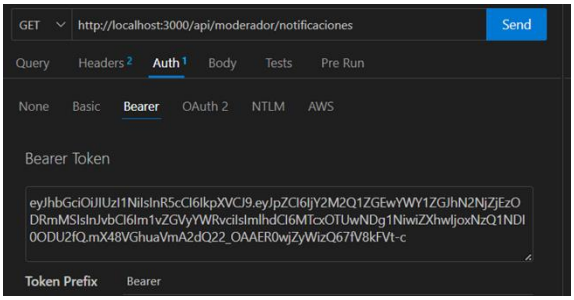


Figura 83: Prueba de Funcionalidad Listar Notificaciones Moderador

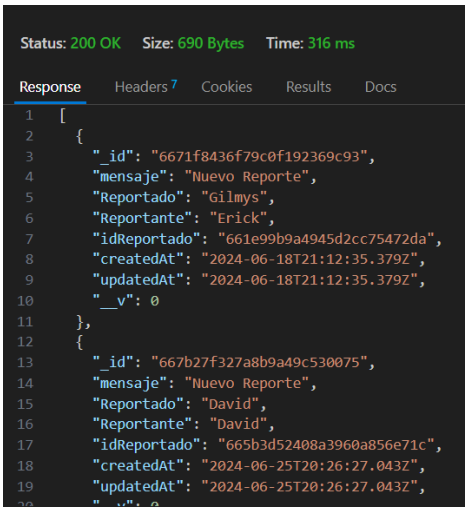


Figura 84: Resultado Prueba de Funcionalidad

PRUEBA DE CARGA

Las pruebas de carga se han realizado mediante el uso de la herramienta Apache JMeter permitiendo evaluar el rendimiento, un total de 23 endpoints relevantes para la aplicación se han sometido a una carga de 50 usuarios por endpoint, dando un total de 1150 peticiones al servidor desplegado. Las configuraciones y resultados obtenidos se visualizan en las **Figura 85** a la **Figura 90**.

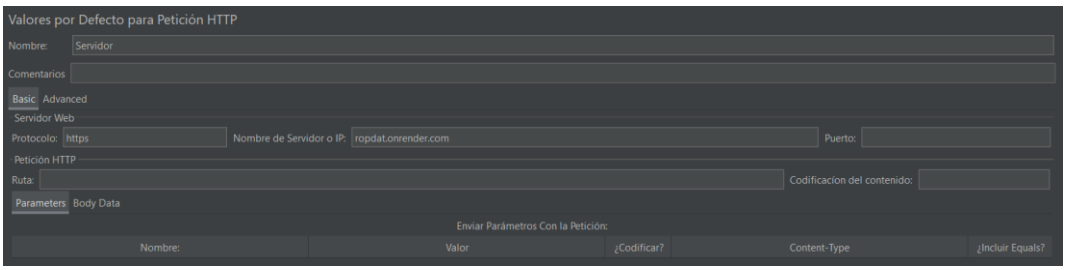


Figura 85: Configuración del servidor en JMeter

Figura 86: Configuración del grupo de hilos (usuarios) en JMeter

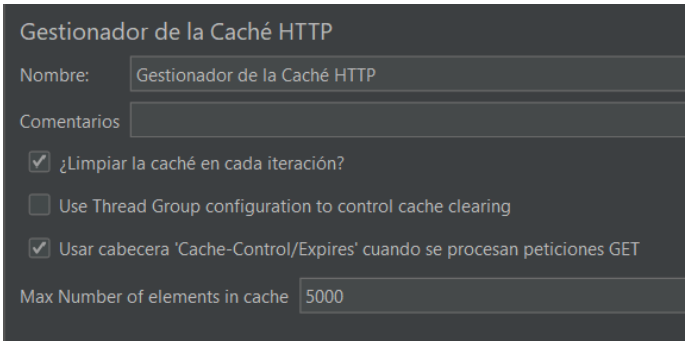


Figura 87: Configuración del Cache de las peticiones

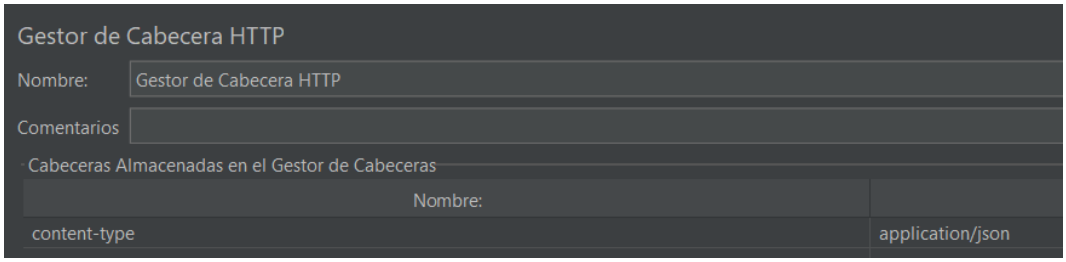


Figura 88: Configuración de la Cabecera de las peticiones

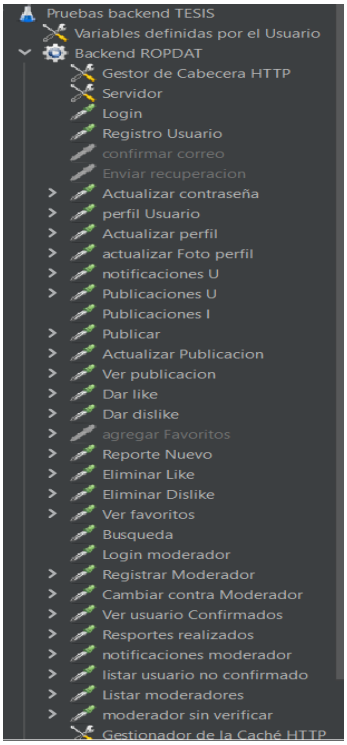
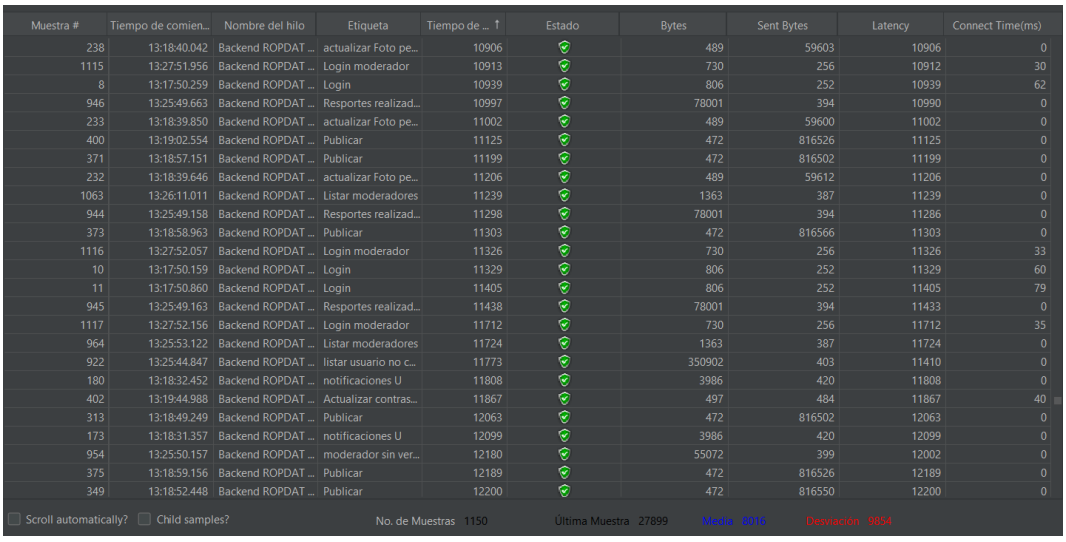


Figura 89: Configuraciones de las solicitudes HTTPS



Muestra #	Tiempo de comien...	Nombre del hilo	Etiqueta	Tiempo de ... ↑	Estado	Bytes	Sent Bytes	Latency	Connect Time(ms)
238	13:18:40.042	Backend ROPDAT ...	actualizar Foto pe...	10906		489	59603	10906	0
1115	13:27:51.956	Backend ROPDAT ...	Login moderador	10913		730	256	10912	30
8	13:17:50.259	Backend ROPDAT ...	Login	10939		806	252	10939	62
946	13:25:49.663	Backend ROPDAT ...	Resportes realizad...	10997		78001	394	10990	0
233	13:18:39.850	Backend ROPDAT ...	actualizar Foto pe...	11002		489	59600	11002	0
400	13:19:02.554	Backend ROPDAT ...	Publicar	11125		472	816526	11125	0
371	13:18:57.151	Backend ROPDAT ...	Publicar	11199		472	816502	11199	0
232	13:18:39.646	Backend ROPDAT ...	actualizar Foto pe...	11206		489	59612	11206	0
1063	13:26:11.011	Backend ROPDAT ...	Listar moderadores	11239		1363	387	11239	0
944	13:25:49.158	Backend ROPDAT ...	Resportes realizad...	11298		78001	394	11286	0
373	13:18:58.963	Backend ROPDAT ...	Publicar	11303		472	816566	11303	0
1116	13:27:52.057	Backend ROPDAT ...	Login moderador	11326		730	256	11326	33
10	13:17:50.159	Backend ROPDAT ...	Login	11329		806	252	11329	60
11	13:17:50.860	Backend ROPDAT ...	Login	11405		806	252	11405	79
945	13:25:49.163	Backend ROPDAT ...	Resportes realizad...	11438		78001	394	11433	0
1117	13:27:52.156	Backend ROPDAT ...	Login moderador	11712		730	256	11712	35
964	13:25:53.122	Backend ROPDAT ...	Listar moderadores	11724		1363	387	11724	0
922	13:25:44.847	Backend ROPDAT ...	listar usuario no c...	11773		350902	403	11410	0
180	13:18:32.452	Backend ROPDAT ...	notificaciones U	11808		3986	420	11808	0
402	13:19:44.988	Backend ROPDAT ...	Actualizar contras...	11867		497	484	11867	40
313	13:18:49.249	Backend ROPDAT ...	Publicar	12063		472	816502	12063	0
173	13:18:31.357	Backend ROPDAT ...	notificaciones U	12099		3986	420	12099	0
954	13:25:50.157	Backend ROPDAT ...	moderador sin ver...	12180		55072	399	12002	0
375	13:18:59.156	Backend ROPDAT ...	Publicar	12189		472	816526	12189	0
349	13:18:52.448	Backend ROPDAT ...	Publicar	12200		472	816550	12200	0

Figura 90: Resultados de las peticiones realizadas

PRUEBAS DE RENDIMIENTO

Las pruebas de rendimiento se realizaron al mismo tiempo que las pruebas de carga en la herramienta Apache JMeter, por lo tanto, se ha analizado el rendimiento del componente ROPDAT con una carga de 50 hilos (usuarios) a los 23 endpoints fundamentales, los resultados obtenidos nos permiten identificar que en dicha carga el componente no presenta fallos y tiempos de carga extensos, presentando un rendimiento esperado. La **Figura 7.91** muestra los resultados obtenidos de las pruebas.

Reporte resumen

Nombre: Rendimiento

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo | Navegar... | Log/Mostrar sólo: | Escribir en Log Sólo Errores | Éxitos | Configurar

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Login	50	22683	4569	41989	11731.22	0.00%	1.1/sec	0.84	0.26	806.0
perfil Usuario	50	13091	3419	40710	11116.61	0.00%	57.6/min	79.03	0.38	84337.0
Actualizar perfil	50	5195	1100	35140	6082.04	0.00%	1.1/sec	0.53	0.59	495.0
actualizar Foto ...	50	13742	4680	28112	5717.48	0.00%	1.0/sec	0.49	59.15	489.0
notificaciones U	50	4236	1581	12099	2548.26	0.00%	1.4/sec	5.34	0.56	3986.0
Publicaciones U	50	3626	1899	9698	1605.17	0.00%	1.4/sec	100.21	0.52	73695.4
Publicaciones I	50	2358	898	5317	1094.05	0.00%	1.5/sec	107.42	0.25	74786.0
Publicar	50	10822	3836	20002	3833.83	0.00%	1.4/sec	0.65	1116.92	472.0
Actualizar contr...	50	36265	9599	56956	15049.61	0.00%	48.7/min	0.39	0.38	497.0
Actualizar Public...	50	1728	941	3912	793.01	0.00%	5.9/sec	2.79	3.35	487.0
Ver publicacion	50	1469	573	3191	842.81	0.00%	5.0/sec	4.20	1.98	861.0
Dar like	50	2312	1307	5007	841.85	0.00%	3.9/sec	1.81	1.66	473.0
Dar dislike	50	2678	1390	4390	836.71	0.00%	3.9/sec	1.83	1.67	476.0
Reporte Nuevo	50	3039	1505	6194	1056.33	0.00%	4.2/sec	1.96	1.99	478.0
Ver favoritos	50	2038	667	6222	1145.03	0.00%	4.5/sec	8.60	1.88	1941.0
Busqueda	50	3595	1045	9304	2018.67	0.00%	3.4/sec	356.58	0.91	108621.0
Ver usuario Conf...	50	3476	772	9566	2415.56	0.00%	2.2/sec	7.59	0.82	3607.0
Resportes realiz...	50	7377	1816	16098	3783.72	0.00%	1.3/sec	101.27	0.51	78001.0
notificaciones m...	50	4910	1177	15104	3586.38	0.00%	1.3/sec	1.12	0.49	915.0
listar usuario no ...	50	10380	3371	18370	5206.19	0.00%	1.2/sec	412.11	0.47	350898.5
Listar moderado...	50	6248	845	16338	4456.52	0.00%	1.3/sec	1.72	0.49	1363.0
moderador sin v...	50	6467	1288	18478	4150.95	0.00%	1.3/sec	70.54	0.51	55072.1
Login moderador	50	16636	3912	27899	7216.35	0.00%	1.5/sec	1.09	0.38	730.0
Total	1150	8016	573	56956	9854.39	0.00%	1.8/sec	64.97	68.13	36673.3

Figura 7.91: Resultados del Rendimiento del Componente ROPDAT

ANEXO III

El siguiente enlace permite visualizar el manual del componente backend ROPDAT desarrollado.

https://www.youtube.com/watch?v=AfnAKCeEWaE

ANEXO IV

Se presenta las credenciales de “prueba” de cada rol existente del backend ROPDAT, y se adjunta el repositorio (Repositorio GitHub) donde se aloja el código fuente del componente.

DOCUMENTACIÓN

A continuación, se presenta el enlace de la documentación realizada a los endpoints del componente backend ROPDAT:

https://www.postman.com/rcee33/workspace/tesis

REPOCITORIO DE GITHUB

https://github.com/Ruizerick26/TESIS

CREDENCIALES

Super Moderador

- **Correo:** moderaro_super@hotmail.com
- **Contraseña:** lg1961s37w

Rol moderador

- **Correo:** moderador@gmail.com
- **Contraseña:** Admin12345

Rol usuario

- **Correo:** usuarioA@gmail.com
- **Contraseña:** UsserA12345