



Manual técnico

Proyecto Profesional I Datacrypt - Equipo 1

Docentes:

- Juan Carlos Monteros
- Francisco Orozco De La Hoz
- Leandro Dikenstein

Integrantes:

- Benitez Micaela - DNI: 43 473 284
- Benitez Yamila - DNI: 41 199 878
- Clauser Nahuel - DNI: 39 803 927
- Gómez Federico - DNI: 40 743 800
- Perez Giannina – DNI: 43 729 769
- Prieto Lucas - DNI: 43 626 494
- Torrico Franco – DNI: 42 370 140

Tabla de contenido

1. Introducción:

- Breve descripción de la API.**
- Objetivos y propósito de la API.**
- Audiencia objetivo del manual.**

2. Requisitos del Sistema:

- Especificaciones de software necesarias para ejecutar la API.**
- Dependencias externas.**

3. Instalación y Configuración:

- Pasos detallados para instalar y configurar la API.**
- Configuración de entornos (desarrollo, prueba, producción).**

4. Despliegue y mantenimiento

- Proceso de Despliegue**
- Mantenimiento Regular**

5. Autenticación y Autorización:

- Métodos de autenticación admitidos.**

6. Casos de Uso:

- Diagrama de Escenarios de uso**

7. Manejo de Errores:

- Descripción de códigos de estado HTTP.**
- Mensajes de error comunes y posibles soluciones.**

8. Problemas Frecuentes:

- Lista de problemas que los usuarios pueden encontrar.**
- Soluciones sugeridas para cada problema.**

9. Seguridad:

- Medidas de seguridad implementadas.**
- Prácticas recomendadas para asegurar la API.**

10. Plan de integración de mejoras a futuro en relación a la experiencia del cliente:

- Propuestas de mejoras futuras.**
- Actualizaciones planeadas.**

11. Soporte Técnico:

- Información de contacto para el soporte técnico.**
- Procedimientos para informar problemas o solicitar ayuda.**

12. Anexos:

- Recursos adicionales.**

1. Introducción:

En el presente manual se definen los aspectos técnicos más relevantes para el adecuado funcionamiento de la aplicación web Datacrypt. Además se describen las herramientas y software utilizados, además de la arquitectura que se definió para el desarrollo del proyecto.

Datacrypt es una aplicación RESTful que puede ser ejecutada en cualquier explorador web, debido a esto no es necesario descargar ningún aplicativo local para su ejecución.

La API, conocida como "Datacrypt API", está diseñada para proporcionar operaciones CRUD sobre Usuarios definidos, diagnósticos y tareas adicionales como gestión de imágenes y contactos. Implementa Flask-RESTx con una documentación integrada en Swagger, mejorando la usabilidad en la interfaz de la api.

La seguridad se maximiza mediante el cifrado AES para datos sensibles y autenticación y autorización en JWT para proteger los endpoints. El objetivo es ofrecer una API segura y eficiente para gestionar usuarios, diagnósticos y tareas complementarias.

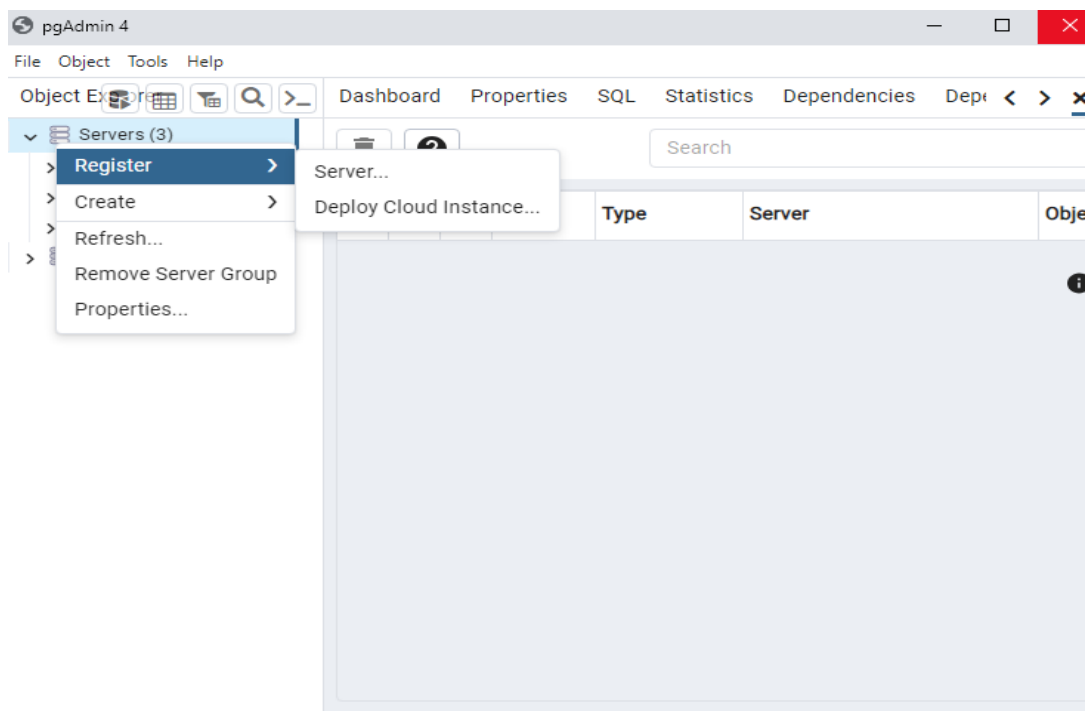
Este manual está dirigido a desarrolladores y cualquier persona que esté involucrada en el mantenimiento, desarrollo o integración del sistema de gestión de usuarios.

- Permite la capacitación de los empleados a través de la descarga del material que se encuentra en la plataforma.
- Permite el registro de condiciones inseguras y accidentes de trabajo.

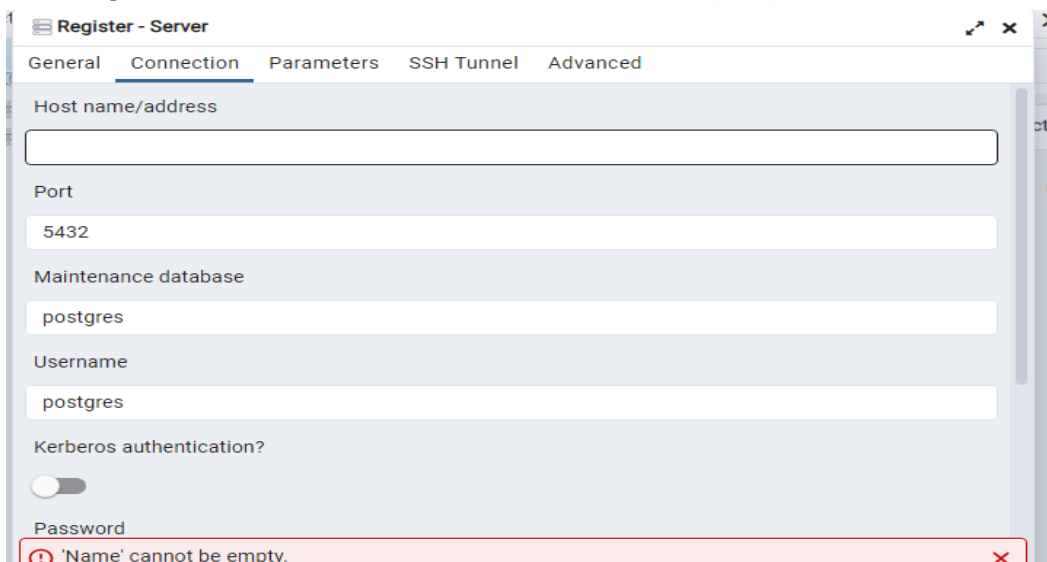
2. Requisitos del Sistema: Aspectos tecnicos

La API requiere de Python 3.6 o superior, y tener instalado en entorno local PostgreSQL v16, luego (recomendado el uso de pgadmin4)
Para los siguientes pasos pedir las credenciales al soporte.

- Descargar e instalar el producto en <https://www.postgresql.org/>
- Descargar e instalar el producto en <https://www.pgadmin.org/download/>
- Uso de pgadmin4 y registro de servidor



- Registro de la base de datos credenciales proporcionadas



y la instalación de dependencias gestionadas por pip en requirements.txt en el entorno local.

La configuración de variables de entorno privadas proporcionada por el equipo, para claves de cifrado AES y cadenas de conexión a la base de datos, son esenciales para un funcionamiento adecuado. Se deberá solicitar las claves al soporte o asignar manualmente las futuras claves en el entorno privado de render.

Para alojar nuestros servicios utilizamos Render como nube gratuita, y base de datos que se proporciona en el mismo servicio de render. También se debe contar con la última actualización del repositorio en github que conecta a los servicios de render.

3. Instalación y Configuración (entendiendo la implementación):

La instalación y configuración se facilitan clonando el repositorio desde <https://github.com/LucasPrieto30/ApiResultados>. Incluyendo la instalación de dependencias, configuración de variables de entornos privados necesarias para el cifrado y las conexiones. En cuando se de un commits el deploy es automático en Render al estar conectado directo al repositorio.

| | | | | | | | | | |
|--------------------------|----------|--|-------------|----------|-----------|------------|------------|------|-------|
| render | | | | | Dashboard | Blueprints | Env Groups | Docs | New + |
| Overview | | | | | | | | | |
| Q Search services | | | | | | | | | |
| Active 7Suspended 0All 7 | | | | | | | | | |
| NAME | STATUS | | TYPE | RUNTIME | REGION | | | | |
| pruebalogin | Deployed | | Web Service | Python 3 | Frankfurt | | | | |

ejemplo de autodeploy

4. Despliegue y mantenimiento

La herramienta pgAdmin nos permite gestionar de una manera simple los Backups, generando un archivo de respaldo de la base de datos con las opciones de configuración que apliquemos. Por esta razón, pgAdmin es donde llevaremos a cabo este proceso. Los backups se realizan **semanalmente**, para conservar la consistencia de los datos.

Por otro lado, para esta instancia estaremos realizando un Restore de prueba con la misma herramienta, con la cual crearemos una nueva base de datos y llevaremos a cabo el proceso de restauración a partir del archivo de backup generado anteriormente.

Estos proceso se pueden llevar a cabo mediante la interfaz de pgAdmin o mediante los siguientes comandos:

Backup: `pg_dump -U usuario -W -h host basename > basename.sql`

Restore: `psql -U username -W -h host basename < basename.sql`

En donde:

U se refiere al usuario propietario de la base de datos o el usuario postgresQL.

W para solicitar el password del usuario antes especificado.

H para indicar cuál es el servidor PostgreSQL

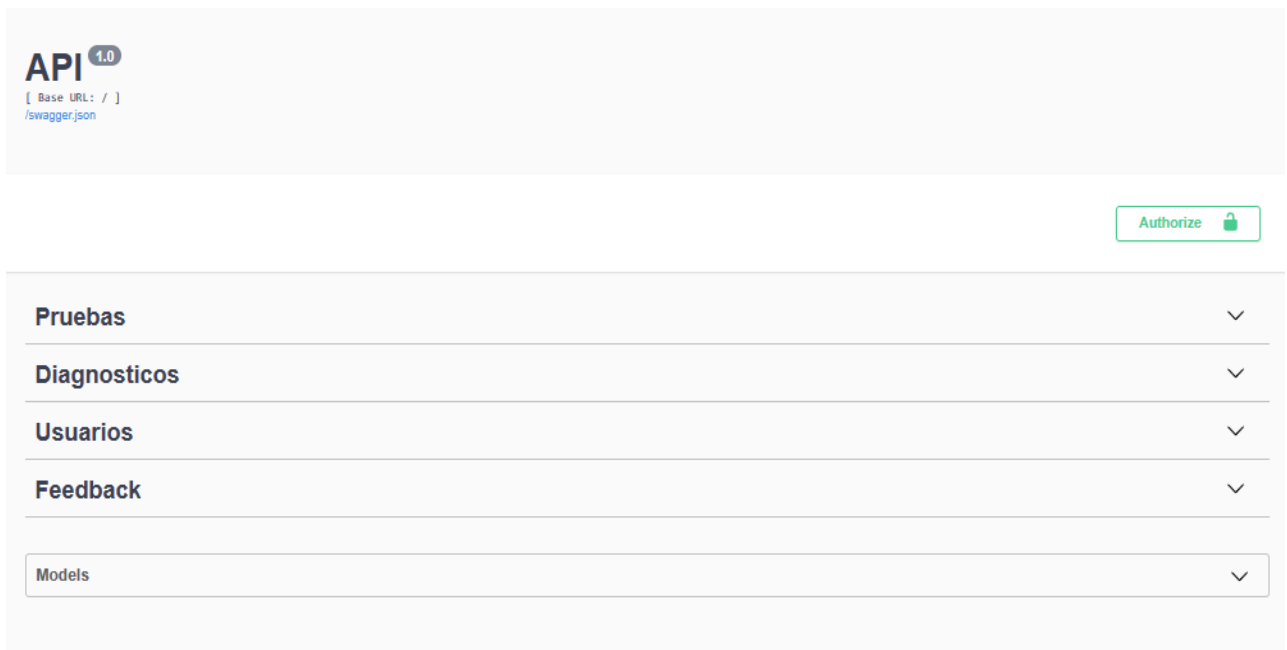
local: localhost

remota: IP del servidor PostgreSQL

El mantenimiento se realiza mediante Servicios como render o utilizando servicios en la nube para el guardado del backup. El mantenimiento incluye la aplicación de actualizaciones, verificar el uso de la seguridad y la supervisión continua del rendimiento.

El despliegue de la api documentación en línea se verá a continuación:

- Vía url [API \(api-resultados.onrender.com\)](https://api-resultados.onrender.com) o entorno local tendrá la interfaz siguiente.

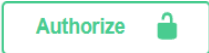


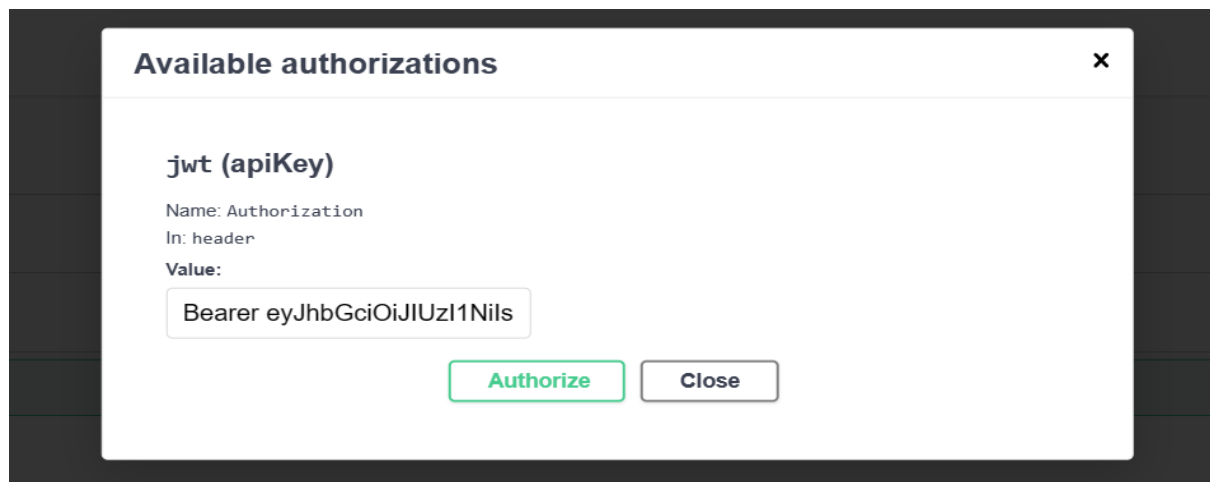
5. Autenticación y Autorización para uso de los endpoints:

La autenticación se realiza mediante JWT para endpoints sensibles, con roles de usuario (administrador, Médico, Auditor y profesional de la salud) determinando los permisos.

La autenticación presente en la api en el endpoint /login se obtiene el token de JWT el cual incluye dentro la información del usuario.

El usuario proporciona sus credenciales dni y password. Para que luego el servidor verifique las credenciales. Si son válidas, el servidor genera un JWT y lo devuelve al cliente.

El cliente almacena el JWT, generalmente en el almacenamiento local, en este caso para el uso del jwt cuando se inicia sesión exitosamente en Usuario/login se devuelve un token el cual se ingresa en  en la casilla value: Bearer <token>



Inclusión del JWT en las Solicitudes:

De esta forma se pueden usar los endpoints con candado, en caso de no tener el token forzado sin el previo paso de login, en aquellos endpoints cerrados devolverá **jwt token faltante** y no será posible el uso. La autorización de estos se presenta en los roles guardados en el JWT.

6. Casos de Uso:

Actores identificados: Medico, Auditor, Administrador, Profesional de la salud, y usuario, este último es previo a la autenticación.

Los actores son los diferentes roles que pueden interactuar con el sistema.

En este caso, los actores son:

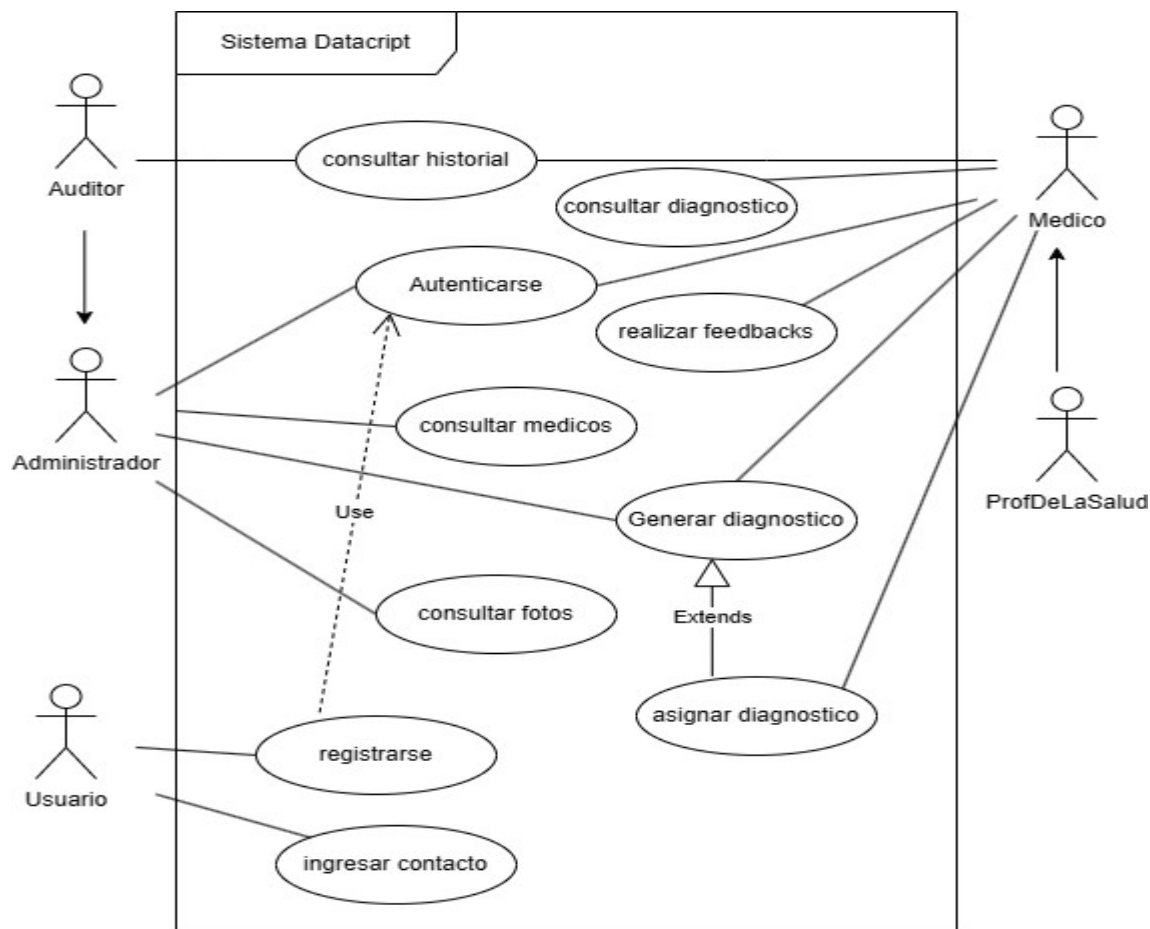
Usuario: Una persona que utiliza el sistema para obtener información o realizar un registro mismo y único.

Médico: Un profesional de la salud que utiliza el sistema para diagnosticar y tratar a los pacientes.

Administrador: Un profesional que utiliza el sistema para gestionar el funcionamiento del sistema.

Auditor: El auditor es un profesional que se encarga de verificar la calidad del sistema. Los casos de uso que podría realizar el auditor son:

Profesional de la salud: El profesional de la salud es un profesional que no es médico, pero que sí trabaja en el campo de la salud. Los casos de uso que podría realizar el profesional de la salud.



7. Manejo de Errores:

La API está configurada y responde con códigos de estado HTTP estándar.

- 200 OK: Operación exitosa.
- 400 Bad Request: Solicitud incorrecta.
- 401 Unauthorized: Fallo en la autenticación.
- 404 Not Found: Recurso no encontrado.
- 500 Internal Server Error: Error interno del servidor.

Mensajes de error detallados y sugerencias de solución se incluyen en el cuerpo de la respuesta.

En caso de ser necesario buscar más detalles, verificar los registros de Actividades y logs en la terminal de render para la Información sobre qué eventos se registran y cómo se almacenan los registros. Así como la terminal también en la base de datos principal.

Los modelos externos de machine learning proporcionan sus propios códigos de respuesta en caso de ser errores causados por factores externos. como imágenes y predicciones.

8. Problemas Frecuentes:

Error de Autenticación JWT:

El proceso de desarrollo de software implica numerosos desafíos y consideraciones. Uno de los problemas recurrentes es el error de autenticación JWT, que puede surgir cuando no se proporciona un token válido o cuando se excede el tiempo de expiración, en este caso no está implementado, está especificado en mejora a futuro. La solución radica en garantizar que cada solicitud autenticada incluya un token JWT válido y verificar el correcto desbloqueo de los candados de seguridad.

Conexión con Tiempo de Expiración:

Otro aspecto crítico es la gestión de la conexión, especialmente cuando se enfrenta a tiempos de expiración. Verificar la seguridad de las conexiones y asegurarse de un llamado adecuado a la base de datos son pasos esenciales para prevenir problemas de conexión. Además, en situaciones donde la base de datos principal experimenta fallas, la clave está en la rápida transición al backup de PostgreSQL cargado en Render para mantener la continuidad del servicio.

Caída de la Base de Datos Principal:

En el caso de que se detecte una posible caída de la base de datos principal, se recomienda seguir el siguiente procedimiento.

Descripción del Problema:

Informar sobre la posibilidad de la caída de la base de datos principal, especificando cualquier mensaje de error o síntoma observado.

Solución Propuesta:

Conectar al backup de PostgreSQL cargado en Render para mantener la continuidad del servicio. La conexión al respaldo de la base de datos asegura que el sistema pueda seguir operando sin interrupciones significativas.

Carga Errónea de Claves Privadas:

La carga errónea de claves privadas puede ser una fuente de problemas, inutilizando conexiones a la base de datos y cifrados. La solución implica una carga cuidadosa y correcta de estas claves, subrayando la importancia de la precisión en los entornos locales.

No Especificar Nuevas Librerías en requirement.txt:

La gestión de dependencias también es crítica para un desarrollo eficiente. La falta de especificación de nuevas librerías en el archivo requirement.txt puede llevar a conflictos y problemas de compatibilidad. Ejecutar el comando 'pip freeze > requirements.txt' en la terminal es una práctica recomendada para listar y especificar las librerías necesarias en el entorno virtual, facilitando así la reproducción del entorno en diferentes instancias.

Problema al No Cargar Imágenes:

En el ámbito de procesamiento de imágenes, problemas al cargarlas pueden surgir debido a formatos incorrectos o dimensiones inadecuadas. Al revisar que las imágenes sean de los formatos adecuados (png, jpg, jpeg) y tengan el tamaño correcto (224x224), se puede mitigar este problema común.

Error en Llamados a Modelos en /feedbacks y /predecir:

Finalmente, los errores en los llamados a modelos, especialmente en las rutas /feedbacks y /predecir, pueden afectar significativamente la funcionalidad del sistema. Mantenerse actualizado y verificar posibles cambios en la documentación es esencial para garantizar una integración fluida con los modelos presentados en la sección **12**.

Anexos: Referencias a Recursos Externos. La adaptabilidad continua es clave para evitar interrupciones no deseadas en el flujo de trabajo del software. Para entender los errores se debe volver a los códigos de respuesta en **7. Manejo de Errores**

Problemas de Agotamiento de Recursos del Servidor

Por otro lado, Problemas de Agotamiento de Recursos del Servidor: El sistema puede enfrentar problemas debido al agotamiento de recursos del servidor, como CPU o memoria, lo que resulta en ralentización o bloqueo del servicio.

Solución: Monitorear y optimizar el uso de recursos, implementar estrategias de escalabilidad y considerar la posibilidad de migrar a un entorno con mayores recursos.

Gestión de Pérdida de Contraseña: Endpoint de Recuperación

En el caso de pérdida de contraseña, por parte del usuario con rol definido. Lo cual genera un problema. Se solicita al servidor la recuperación de este, para la solución se presenta un código de chequeo vía email. El proporcionado durante el registro para validación de código OTP. La respuesta de validación de un token de uso único, el cual se envía mediante header en el endpoint /reset_passwod. Este circuito es de único uso. En caso de escribir mal la contraseña o perder el token, se deberá realizar el circuito de solicitud al servidor nuevamente,

9. Seguridad:

En términos de seguridad de información nos regimos bajo el marco de la Ley Nacional de Protección de Datos Personales (Ley 25.326). El tratamiento de datos personales, incluida la información médica. Entre otras cosas, establece principios de protección de datos, define los derechos de los titulares de la información y exige el consentimiento informado.

- También se especifica la Comunicación segura mediante HTTPS
- Datos sensibles cifrados usando AES-128, no se admiten otros.
- Validación y sanitización de todas las entradas de usuario antes de guardar resultados en la DB.
- Contraseñas almacenadas de manera segura mediante funciones de hash. Cada contraseña que ingrese debe cumplir con ciertas políticas de

caracteres. `^(?!.*(.*)1)(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[~!@#$%^&*()_+!-=\{\}\|\[\]\\\\\.;'<>?])[A-Za-z0-9~!@#$%^&*()_+!-=\{\}\|\[\]\\\\\.;'<>?]{8,}$'`

- ^: Indica el inicio de la cadena.

- Deberá ejecutar la política de “Las contraseñas deberán ser cambiadas cada 60 días por seguridad, por lo tanto, implementamos una función que permite verificar si es necesario cambiar una contraseña en función de cuándo fue la última vez que se cambió”.

Al cumplimentar esto el ingreso de nuevas claves deben ser solicitadas en la sección soporte técnico de cómo implementar las nuevas.

Se deberá ejercer en cada actualización la evaluación de Vulnerabilidades en el Sistema: Se utilizará la herramienta OWASP ZAP (Zed Attack Proxy) para probar y evaluar las vulnerabilidades en el sistema. En la sección **12. Anexos Enlaces externos** dejaremos el reporte generado por ZAP.

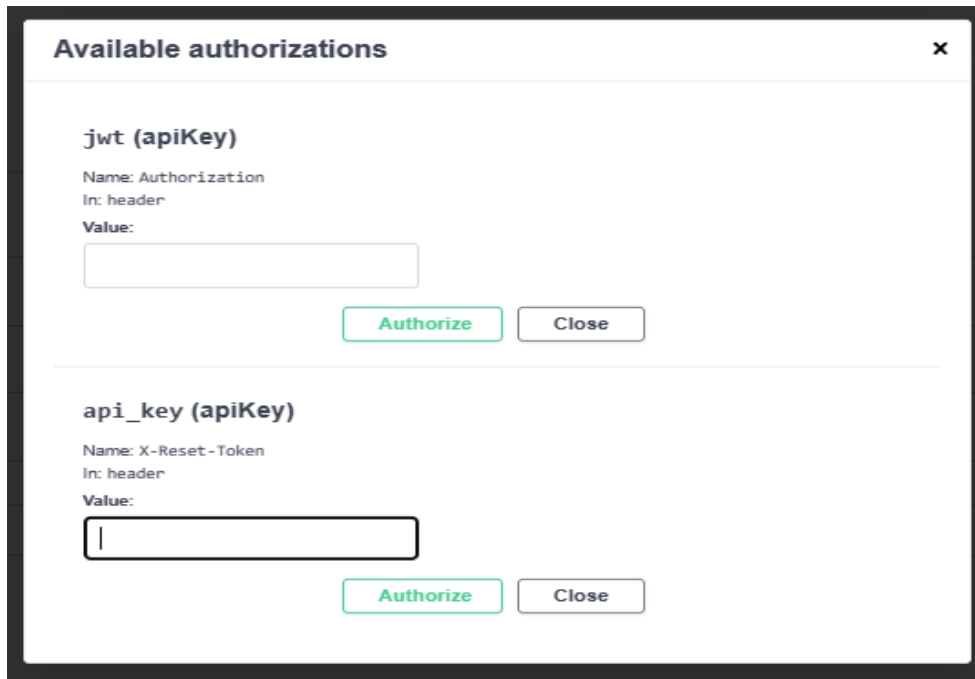
10. Plan de integración de mejoras a futuro en relación a la experiencia del cliente:

Explorar la integración de nuevas tecnologías y actualizaciones para mejorar la experiencia del usuario.

Evaluar la posibilidad de agregar nuevas características para abordar las necesidades emergentes de los usuarios.

- La incorporación de nuevas funcionalidades como los contactos y el manejo de estas solicitudes, así como nuevos modelos para predecir y realizar feedbacks tiene un crecimiento lineal.
- Las tecnologías y herramientas nos permiten realizar un plan de integración y uso de machine learning para nuevos modelos de predicciones.
- En cuanto a la mejora de interfaz y la documentación swagger se tiene un plan de mejoras para la visualización de la api. Así como señalar las nuevas api keys que se introduzcan sea el caso para endpoints específicos.

- La incorporación y el manejo de nuevas api keys para introducir a la api y proporcionar nuevas seguridades con jwt es más fácil para futuras implementaciones.



Available authorizations [X]

jwt (apiKey)
 Name: Authorization
 In: header
 Value:
 [Authorize] [Close]

api_key (apiKey)
 Name: X-Reset-Token
 In: header
 Value:
 [Authorize] [Close]

Mejoras en la eficiencia y escalabilidad en recursos render a través de la monetización.

11. Soporte Técnico:

Gestión Efectiva de Problemas y Consultas

Para garantizar una experiencia fluida y eficiente, nuestro equipo de soporte técnico está dedicado a brindar asistencia integral. Si requiere información adicional o asistencia, no dude en ponerse en contacto con nosotros a través de la dirección de correo electrónico **datacrypt2023@gmail.com**. También puede acceder a nuestra documentación en línea en <https://api-resultados.onrender.com/> para obtener recursos útiles.

Procedimientos para Reportar Problemas:

En el caso de que se encuentre con algún problema, le recomendamos seguir estos pasos al informarnos mediante **datacrypt2023@gmail.com**

Descripción Detallada del Problema:

- Proporcione una descripción detallada del problema que está experimentando. Cuanta más información podamos obtener sobre la naturaleza del problema, más rápido y preciso será nuestro proceso de resolución.
- Capturas de Pantalla:

Si es posible, adjunte capturas de pantalla que ilustren el problema. Las imágenes pueden ser herramientas valiosas para comprender visualmente los desafíos que pueda estar enfrentando, permitiéndonos abordar el problema de manera más efectiva.

- Información del Sistema Operativo y Versión del Software:

Proporcione detalles sobre su sistema operativo y la versión del software que está utilizando. Esta información es crucial para entender el entorno en el que se está produciendo el problema y facilita el proceso de diagnóstico.

- Compromiso con la Resolución Rápida:

Nos comprometemos a abordar cualquier problema que pueda surgir de manera oportuna y eficiente. Su retroalimentación es fundamental para mejorar continuamente nuestros servicios, y nuestro equipo de soporte está aquí para garantizar que obtenga la asistencia necesaria.

- Agradecemos su colaboración y paciencia mientras trabajamos juntos para resolver cualquier inconveniente. Su satisfacción y éxito son nuestra prioridad, y estamos aquí para respaldarlo. **¡Gracias por confiar en nosotros!**


12. Anexos: Recursos Adicionales:

Enlaces a Documentación Externa:

- Se presenta un plan de mejoras de api keys y documentación
<https://pruebalogin.onrender.com/>

Referencias a Recursos Externos:

- Implementación de JWT
<https://docs.google.com/document/d/1mzoP23v7TN-4vEw7gjiF4wKxzgdgay2hpmfbYSGMLP4/edit>

- Posibles errores de jwt
https://docs.google.com/document/d/1_ncr8lPflrY03xDK7ufvjE3ODioZh01G2nj7SH_QoXM/edit
- Diagrama de la BBDD relacionada y función de los endpoints de predicción y feedbacks
 Documentación BBDD
- Validación último Reporte a la fecha generado por ZAP en la api
https://drive.google.com/drive/folders/1UhBuhShZuKLcb6Qn80o7P_sNeb1zhoQ