| SCRIP DE LA BBDD |
| --- |
| **Proyecto:** “Plataforma de Gestión de Servicios y Agenda Profesional” |
|  |
|  |
| Versión*: 1.5* |
| 06/11/2024 |

# Contenido

[**1. Contenido 1**](#_heading=h.d2xiuc4dl1oj)

[**2. Ficha del documento 2**](#_heading=h.gjdgxs)

[2.1. Versionamiento 2](#_heading=h.30j0zll)

[2.2. Integrantes 2](#_heading=h.1fob9te)

[**3. Introducción 3**](#_heading=h.eifsl56ht4k1)

[3.1. Propósito del Script para la BBDD en Firebase. 3](#_heading=h.evczh7txp4ve)

[3.2. Importancia en el proyecto y ventajas de usar NoSQL. 3](#_heading=h.ks5v0tj4caxd)

[**4. Definición de Seguridad 3**](#_heading=h.sk4h2m897ip0)

[4.1. Reglas de Seguridad 4](#_heading=h.3nc8w5kpmn66)

[4.2. Autenticación y Autorización 5](#_heading=h.j29qifm7xndg)

[4.3. Control de Acceso 5](#_heading=h.ynjur0ub5lqz)

[**5. Código 6**](#_heading=h.7kpoqbsadpe5)

# Ficha del documento

## Versionamiento

| Versión | Fecha | Modificación |
| --- | --- | --- |
| *1.0* | *11/09/2024* | *Creación inicial del documento: Se define la estructura básica y secciones preliminares, como contenido, ficha del documento, versión y equipo de desarrollo.* |
| *1.1* | *12/09/2024* | *Desarrollo del script básico de la base de datos: Se añade el código inicial para la creación de tablas y estructuras fundamentales en la base de datos.* |
| *1.2* | *16/10/2024* | *Incorporación de relaciones y restricciones: Se añaden relaciones entre tablas y restricciones como claves primarias y foráneas en el script de la BBDD.* |
| *1.3* | *04/11/2024* | *Pruebas y optimización del script: Se realizan pruebas en el entorno de desarrollo para asegurar la funcionalidad del script y se optimizan consultas o estructuras según los resultados obtenidos.* |
| *1.4* | *05/11/2024* | *Revisión técnica y ajustes finales: Revisión por parte de un experto para validar la precisión y optimización del script, realizando ajustes menores en el código según las recomendaciones.* |
| *1.5* | *06/11/2024* | *Entrega Final: subir el documento al repositorio Git del equipo de trabajo. Se registra como versión final en el control de versiones.* |

## Integrantes

| Integrantes del equipo | Rut |
| --- | --- |
| *Jairo Álvarez* | *21.055.328-2* |
| *Francisco López* | *20.883.087-2* |
| *Constanza Painevilo* | *19.561.395-8* |
| *Bastián Rodríguez* | *20.526.431-0* |

**Carrera:** Ingeniería en Informática

**Sede:** San Joaquín

# 

# Introducción

En el desarrollo de proyectos modernos, la gestión de datos segura y eficiente es fundamental para garantizar la usabilidad, integridad y escalabilidad de la solución. Este script está diseñado para implementar y administrar una base de datos en **Firebase**, aprovechando las capacidades de una base de datos **NoSQL** para almacenar y gestionar información relacionada con usuarios, servicios, y su disponibilidad de manera eficiente. Además, el enfoque en la seguridad a través de reglas, autenticación y control de acceso asegura que los datos estén protegidos contra accesos no autorizados.

## Propósito del Script para la BBDD en Firebase.

El propósito principal del script es estructurar y configurar una base de datos en Firebase Firestore para un proyecto que gestione:

* Información de usuarios (nombre, correo electrónico, número de teléfono y ubicación).
* Detalles de servicios ofrecidos (nombre del servicio, profesional encargado, descripción y contacto).
* Disponibilidad del servicio a través de agendas con fechas y horas específicas.

El script implementa una estructura clara y bien definida que facilita la integración con funcionalidades del proyecto, como autenticación de usuarios, control de acceso, y consultas eficientes de datos.

## Importancia en el proyecto y ventajas de usar NoSQL.

La elección de Firebase como base de datos **NoSQL** proporciona varias ventajas que son críticas para el éxito del proyecto:

1. **Escalabilidad y Flexibilidad**: La naturaleza no relacional de Firebase permite almacenar datos jerárquicos y heterogéneos, facilitando la adaptación a cambios en los requisitos del proyecto sin necesidad de reestructurar toda la base de datos.
2. **Integración Sencilla**: Firebase se integra perfectamente con otros servicios como Firebase Authentication, lo que reduce el tiempo de desarrollo.
3. **Sin Backend Complejo**: Firebase elimina la necesidad de configurar y mantener un backend dedicado, ya que gestiona el almacenamiento, consultas y la autenticación de manera nativa.
4. **Tiempo Real**: Las actualizaciones en tiempo real permiten sincronizar datos entre los clientes instantáneamente, una funcionalidad clave para aplicaciones interactivas.
5. **Seguridad Mejorada**: Las reglas de seguridad personalizables aseguran que solo los usuarios autorizados accedan a los datos.

# Definición de Seguridad

La seguridad en Firebase incluye mecanismos para proteger los datos almacenados contra accesos no autorizados. Esto se logra mediante:

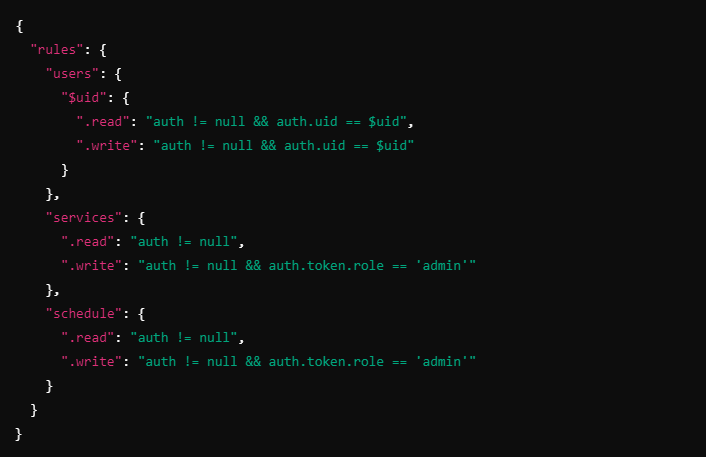
**Reglas de Seguridad**: Configuraciones personalizadas que determinan quién puede leer o escribir datos en la base de datos, basándose en propiedades del usuario autenticado.

**Autenticación**: El proceso de identificar a los usuarios mediante métodos como correo y contraseña, autenticación con Google, o redes sociales.

**Autorización**: Permite aplicar restricciones basadas en roles o atributos del usuario (por ejemplo, "admin", "client").

## Reglas de Seguridad

Las reglas de seguridad en Firebase Firestore definen qué usuarios pueden acceder a qué datos, y bajo qué condiciones. A continuación, un ejemplo básico de reglas:



Solo los usuarios autenticados pueden leer y escribir en sus propios documentos de la colección users.

Solo los administradores (role == 'admin') pueden crear o modificar servicios y agendas.

## Autenticación y Autorización

**Autenticación**: Firebase Authentication facilita la identificación de usuarios mediante múltiples métodos:

* Correo electrónico y contraseña.
* Redes sociales como Google o Facebook.
* Tokens personalizados para autenticación avanzada.

**Autorización**: Una vez autenticado, el sistema asigna permisos según el rol del usuario (por ejemplo, "client" o "admin"). Esto asegura que los usuarios solo accedan a los recursos permitidos.

## Control de Acceso

El control de acceso en Firebase combina:

**Reglas de Seguridad**: Determinan quién puede acceder a qué datos.

**Roles y Perfiles**: Se asignan roles a los usuarios (administrador, cliente, etc.) para segmentar las funcionalidades disponibles.

**Autenticación en Tiempo Real**: Verifica dinámicamente la identidad del usuario antes de permitir el acceso a datos.

# Código

