| EVIDENCIA DEL DESARROLLO DEL SISTEMA |
| --- |
| **Proyecto:** “Plataforma de Gestión de Servicios y Agenda Profesional” |
|  |
|  |
| Versión*: 1.5* |
| 15/11/2024 |

# 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 a la Evidencia de Desarrollo 3**](#_heading=h.vxir0y6sb9vo)

[3.1. Configuraciones Iniciales 3](#_heading=h.qals65b5d3pj)

[3.2. Organización del Código 3](#_heading=h.51e0nsdm90h0)

[3.3. Pruebas y Control de Calidad 4](#_heading=h.rb3ypvmhozm6)

[3.4. Flujo de Trabajo 4](#_heading=h.xvlb74ha19uu)

[**4. Entorno de Desarrollo 4**](#_heading=h.jto2lr21j9r7)

[4.1. Descripción del Entorno 4](#_heading=h.djwtvwqg8b9c)

[4.2. Configuraciones Iniciales 5](#_heading=h.vv3eq3jxw2h)

[4.3. Descripción de los Módulos Desarrollados 5](#_heading=h.5lla7fvy0clr)

[4.3.1. Listado de Módulos 5](#_heading=h.n8utqj3aec11)

[4.3.2. Detalles de cada Módulo 5](#_heading=h.jyt5q1nsq4l7)

[**5. Pruebas de Funcionalidad 6**](#_heading=h.vv3kn5ey52f)

[5.1. Casos de Prueba Realizados 6](#_heading=h.cdxjx6m0g2t0)

[7.2. Resultados de Pruebas 6](#_heading=h.exrcqd7kuqgh)

[7.3. Logs de Errores y Soluciones Implementadas 6](#_heading=h.2lzlf1gkdmdh)

[**8. Integración de Módulos 6**](#_heading=h.11zucek82mxu)

[8.1 Descripción de la Integración 6](#_heading=h.msy5sk50d02)

[**9. Control de Versiones 7**](#_heading=h.iy4dj3f9vl41)

[9.1 Historial de Cambios Relevantes 7](#_heading=h.oi4u2cl1qzz3)

[**10. Documentación Técnica Asociada 7**](#_heading=h.4cv2l8jwnia5)

[10.1 Documentación en el Código 7](#_heading=h.wzchen18uzel)

[10.2 Referencias a Otros Documentos 7](#_heading=h.sheeemqbctzi)

# 

# Ficha del documento

## Versionamiento

| Versión | Fecha | Modificación |
| --- | --- | --- |
| *1.0* | *25/09/2024* | *Creación del documento: Se define la estructura básica y el índice de contenidos, incluyendo secciones preliminares como "Introducción a la Evidencia de Desarrollo" y "Entorno de Desarrollo".* |
| *1.1* | *02/10/2024* | *Incorporación de la evidencia de desarrollo inicial: Se añaden capturas de pantalla, descripciones de código y configuraciones principales del entorno de desarrollo.* |
| *1.2* | *16/10/2024* | *Revisión y ajustes en la evidencia: Se revisa la coherencia de las capturas y descripciones, realizando correcciones en la redacción y asegurando que la documentación esté alineada con los avances reales del desarrollo.* |
| *1.3* | *21/10/2024* | *Expansión del contenido: Se añaden detalles adicionales de desarrollo, incluyendo evidencia de componentes específicos (frontend, backend) y scripts de configuración, y se realiza una revisión técnica para asegurar precisión.* |
| *1.4* | *23/10/2024* | *Revisión de Calidad y Aprobación: Revisión de calidad final para asegurar consistencia, precisión y cumplimiento de estándares, incluyendo correcciones menores en formato y ortografía.* |
| *1.5* | *15/11/2024* | *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 a la Evidencia de Desarrollo

El desarrollo de este proyecto se llevó a cabo utilizando **Angular** como framework principal, en combinación con **Ionic** y **Capacitor**, para garantizar una integración eficiente de funcionalidades específicas tanto en plataformas móviles como web. Esta elección tecnológica responde a la necesidad de ofrecer una experiencia de usuario optimizada y una base tecnológica robusta, adaptable a diferentes dispositivos y contextos. La arquitectura modular adoptada facilita la escalabilidad del sistema, así como su mantenimiento a largo plazo, permitiendo agregar nuevas funcionalidades sin afectar las existentes.

## 3.1. Configuraciones Iniciales

Desde sus etapas iniciales, el proyecto fue configurado con herramientas y tecnologías fundamentales para garantizar un entorno de desarrollo eficiente y organizado. Las configuraciones iniciales incluyen:

* **TypeScript:** Utilizado para proporcionar un tipado estático que mejora la legibilidad y reduce errores durante el desarrollo.
* **ESLint:** Herramienta que asegura estándares de codificación consistentes, reduciendo la posibilidad de errores comunes y mejorando la calidad del código.
* **Karma:** Framework de pruebas unitarias para verificar la funcionalidad y confiabilidad de los componentes y servicios implementados.
* **Firebase:** Plataforma backend utilizada para la gestión de servicios clave, como autenticación, base de datos en tiempo real y almacenamiento.

Estas configuraciones están respaldadas por archivos de configuración específicos como:

* **angular.json**: Define las rutas, configuraciones de compilación y optimizaciones para diferentes entornos.
* **ionic.config.json**: Gestiona la integración con Ionic para garantizar la compatibilidad multiplataforma.
* **capacitor.config.ts**: Configura la interacción con funcionalidades nativas en dispositivos móviles.

## 3.2. Organización del Código

La organización del código sigue un diseño modular que respalda el principio de separación de responsabilidades, asegurando que cada parte del sistema sea independiente y fácilmente reutilizable. Los principales módulos del proyecto son:

* **Core Module:** Contiene servicios globales como la autenticación, utilidades generales y configuraciones que son comunes a toda la aplicación.
* **Guards:** Implementados para proteger el acceso a rutas específicas, validando el estado de autenticación del usuario y garantizando la seguridad del sistema.
* **Pages y Components:** Representan las diferentes interfaces de usuario, organizadas de manera que faciliten la navegación y la interacción del usuario con el sistema.
* **Shared Module:** Incluye componentes reutilizables y estilos compartidos, promoviendo la consistencia visual y funcional en toda la aplicación.

Además, el uso de un sistema de enrutamiento dinámico facilita la navegación fluida entre las distintas páginas y módulos.

## 3.3. Pruebas y Control de Calidad

El control de calidad y las pruebas son pilares fundamentales en este proyecto. Las pruebas unitarias, configuradas en **karma.conf.js**, se ejecutan para validar la funcionalidad de los componentes y servicios desarrollados. Estas pruebas aseguran que cada parte del sistema cumpla con los requisitos funcionales definidos.

Adicionalmente, las reglas establecidas en **.eslintrc.json** garantizan que el código escrito siga estándares claros y consistentes, reduciendo la probabilidad de errores en el desarrollo y facilitando la colaboración entre desarrolladores.

## 3.4. Flujo de Trabajo

El desarrollo del proyecto se realizó utilizando una metodología ágil, enfocada en ciclos iterativos de diseño, desarrollo y validación. Esta metodología permitió:

* Entregas incrementales de funcionalidades, priorizando aquellas de mayor valor para los usuarios.
* Pruebas continuas de nuevas implementaciones para asegurar su correcta integración.
* Flexibilidad para adaptarse a cambios en los requisitos durante el desarrollo.

Cada módulo y servicio fue diseñado para ser independiente, lo que no solo facilita la integración de nuevas funcionalidades, sino también la adaptación del sistema a futuros cambios tecnológicos o de negocio. Este enfoque modular asegura que el sistema pueda escalar de manera efectiva sin comprometer su estabilidad.

# Entorno de Desarrollo

El proyecto fue desarrollado en un entorno moderno y bien estructurado, que combina herramientas y frameworks de última generación para garantizar un flujo de trabajo ágil y eficiente. A continuación, se describen los principales elementos del entorno de desarrollo utilizados:

## Descripción del Entorno

El desarrollo de este proyecto se apoyó en herramientas, lenguajes de programación, frameworks y tecnologías modernas que garantizan un entorno flexible, robusto y orientado a la entrega continua de valor. Estas tecnologías fueron seleccionadas para abordar de manera eficiente los desafíos planteados en los requisitos del sistema:

* **Lenguajes de Programación:**
  + **TypeScript:** Base del desarrollo del frontend, aportando tipado estático que facilita la detección de errores durante la etapa de desarrollo y mejora la calidad del código. Esto asegura un sistema más mantenible y legible.
  + **HTML y SCSS:** Utilizados para construir interfaces de usuario modernas y responsivas, con un diseño visual atractivo y adaptado a diferentes tamaños de pantalla.
* **Frameworks:**
  + **Angular:** Framework principal que proporciona una estructura modular y organizada para el desarrollo, permitiendo la escalabilidad y el mantenimiento del sistema. Sus características, como la inyección de dependencias y el sistema de enrutamiento, optimizan el flujo de desarrollo.
  + **Ionic y Capacitor:** Integrados para extender la funcionalidad de la aplicación hacia entornos híbridos, facilitando la creación de una experiencia uniforme en plataformas móviles y web.
* **Herramientas de Desarrollo:**
  + **Visual Studio Code:** IDE seleccionado por su extensibilidad y facilidad de uso. Se configuraron extensiones específicas para Angular, TypeScript e Ionic, optimizando el desarrollo y la depuración.
  + **Firebase:** Backend serverless utilizado para gestionar la autenticación y el almacenamiento de datos en tiempo real. Su integración permitió reducir la complejidad del desarrollo backend.
  + **Node.js:** Entorno de ejecución que permite gestionar las dependencias del proyecto y ejecutar scripts automatizados, como la compilación y las pruebas.
* **Control de Versiones:**
  + **Git:** Sistema distribuido de control de versiones que permitió un manejo eficiente de cambios en el código, evitando conflictos y facilitando la colaboración entre desarrolladores.
  + **GitHub:** Plataforma utilizada para el alojamiento del repositorio, seguimiento de issues y organización del proyecto mediante ramas específicas para cada funcionalidad.

## Configuraciones Iniciales

Las configuraciones iniciales fueron fundamentales para establecer un entorno de desarrollo consistente y optimizado:

* **Configuración del IDE:** Visual Studio Code fue personalizado con extensiones como **Angular Essentials** y **ESLint**, que garantizan la detección temprana de errores y aseguran que el código cumpla con estándares de calidad definidos.
* **Configuración de Firebase:** La integración inicial incluyó la configuración de un proyecto en Firebase Console, habilitando la autenticación con correo electrónico y el almacenamiento de datos en Firestore. Estas configuraciones se documentaron en los archivos de entorno del proyecto.
* **Configuración de Capacitor:** En el archivo capacitor.config.ts se definieron las plataformas objetivo (Android, iOS y web) junto con los ajustes necesarios para la integración con plugins nativos, como el manejo de cámara y almacenamiento local.

## Descripción de los Módulos Desarrollados

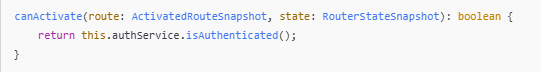
## Listado de Módulos

* **Core Module:** Contiene servicios globales, como la autenticación de usuarios, utilidades de manejo de datos y configuraciones compartidas entre módulos.
* **Shared Module:** Almacena componentes reutilizables (como botones personalizados y cuadros de diálogo) y estilos compartidos para garantizar la consistencia visual en toda la aplicación.
* **Pages Module:** Contiene las diferentes vistas que interactúan directamente con los usuarios, organizadas en secciones específicas de la aplicación.

## Detalles de cada Módulo

**Core Module:**

* **Funcionalidades Clave:**
  + Guardias de rutas (AuthGuard, NoAuthGuard) para proteger el acceso según el estado de autenticación del usuario.
  + Servicios diseñados para interactuar con Firebase, gestionando la autenticación y operaciones básicas sobre Firestore.
* **Fragmento de Código Ejemplar:**typescript  
  Copiar código



**Shared Module:**

* Proporciona componentes como encabezados, botones, y estilos globales que promueven la consistencia y reutilización en toda la aplicación.

# Pruebas de Funcionalidad

## Casos de Prueba Realizados

* Verificación de que los usuarios no autenticados son redirigidos a la pantalla de inicio de sesión.
* Validación de la correcta creación y lectura de datos en Firebase Firestore.
* Pruebas unitarias para confirmar el funcionamiento de los componentes visuales clave.

## 7.2. Resultados de Pruebas

Las pruebas realizadas confirmaron la funcionalidad esperada de cada componente, servicio y módulo, asegurando su cumplimiento con los requisitos. Los resultados de estas pruebas se registraron y revisaron como parte del flujo de trabajo ágil.

## 7.3. Logs de Errores y Soluciones Implementadas

* **Problema Identificado:** Expiración de tokens de autenticación sin manejo adecuado, resultando en errores de acceso.
* **Solución Implementada:** Desarrollo de un interceptor HTTP que detecta la expiración del token y lo renueva automáticamente.

# 8. Integración de Módulos

## 8.1 Descripción de la Integración

Los módulos están conectados a través del sistema de enrutamiento de Angular, que facilita la navegación y la interacción entre vistas. El Core Module actúa como el núcleo funcional, proveyendo servicios y datos a las páginas de la aplicación, mientras que el Shared Module se asegura de que los elementos visuales se mantengan consistentes.

# 9. Control de Versiones

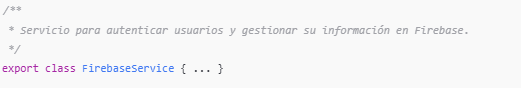
## 9.1 Historial de Cambios Relevantes

* **Commit Clave:** "Implementación de guardias de autenticación". Este cambio añadió la funcionalidad de control de acceso según el estado del usuario, mejorando la seguridad del sistema.

# 10. Documentación Técnica Asociada

## 10.1 Documentación en el Código

Cada servicio, componente y módulo incluye documentación detallada en el código, describiendo su propósito, parámetros y funcionamiento. Por ejemplo:



## 10.2 Referencias a Otros Documentos

La especificación de requisitos y los diagramas de arquitectura están documentados en el **ERS (Especificación de Requisitos del Sistema)**, con un enfoque en la autenticación y seguridad.