

Servihub



CONCEPTO:
OLA DE
SERVICIOS

REFRESCANTE
AZUL
SERVICIOS
ROMPER
EMPUJE
FUERZAS
IR Y VOLVER

- Por qué
- Cómo
- Quién
- Dónde
- Cuándo

¿Por qué?

En 2023, la economía gig generó aproximadamente 204 mil millones de dólares en volumen bruto global, con un crecimiento proyectado del 17% anual compuesto, alcanzando unos 455 mil millones de dólares para 2023.

La economía gig ofrece diversas oportunidades de ingresos. Por ejemplo, el 20% de los trabajadores independientes a tiempo completo gana más de 100,000 dólares anualmente.

El uso de aplicaciones gig es particularmente popular entre las generaciones más jóvenes, con el 38% de los trabajadores de gig economy teniendo entre 18 y 34 años.

Además, un 76% de los trabajadores gig se declara muy satisfecho con su elección, prefiriendo esta modalidad de trabajo sobre los empleos tradicionales debido a la libertad y la satisfacción personal que encuentran en su trabajo

Entonces ...

Por un lado tenemos el problema de que existen numerosas aplicaciones de servicios. Pero todas se concentran en un usuario o servicio específico.

- Transporte y Entrega: Uber, Lyft, DoorDash, Uber Eats.
- Servicios Profesionales y Freelance: Upwork, Fiverr.
- Alojamiento y Compartición de Bienes: Airbnb, Turo.
- Servicios de Cuidado y Reparaciones: TaskRabbit, Handy.
- Otros Gig y reservas: Booksy, Fresha, MindBody, Yelp

El segundo problema es que las aplicaciones de servicios se canibalizan entre ellas provocando una peor calidad de servicios y de precios

Servuhib es una respuesta a esos problemas. Crear una simbiosis entre gente emprendedora y las necesidades que puede tener un usuario

- Usuarios tienen una necesidad
- Las necesidades se solventan con trabajo
- Trabajos reporta beneficios
- Beneficios impactan positivamente en nosotros



En resumen, las apps de servicios en la economía gig han visto un crecimiento robusto y ofrecen una variedad de oportunidades tanto para ingresos a tiempo completo como para trabajos secundarios.

ANÁLISIS DAFO

DEBILIDADES

- Falta de beneficios laborales
- Volatilidad de ingresos
- Desafíos tecnológicos
- Competencia intensa

AMENAZAS

- Crisis económica
- Regulación y cumplimiento legal
- Cambio en las preferencias del consumidor
- Riesgos de seguridad y privacidad

FORTALEZAS

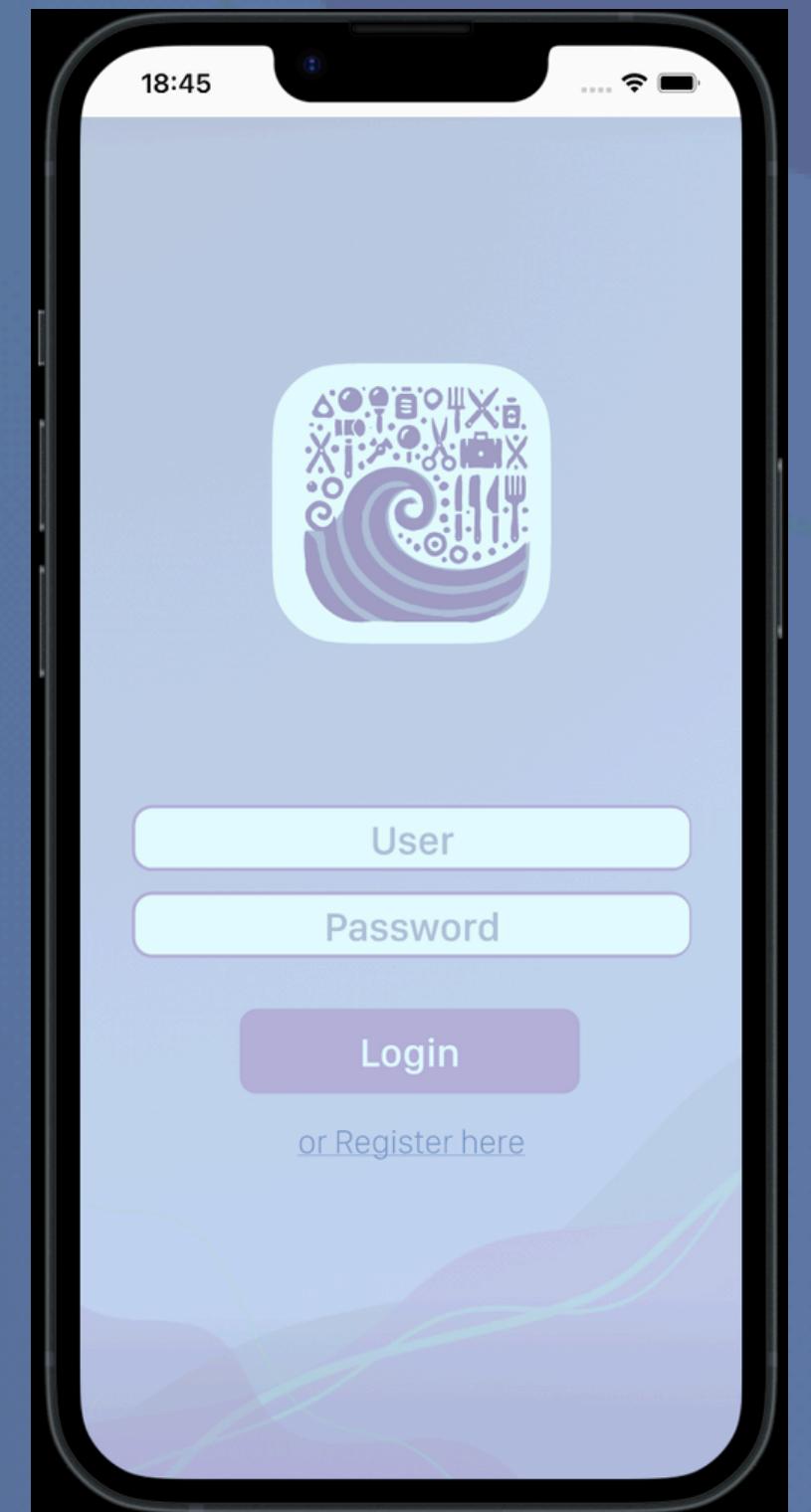
- Flexibilidad y autonomía
- Diversificación de servicios
- Escalabilidad
- Uso de últimas tecnologías

OPORTUNIDADES

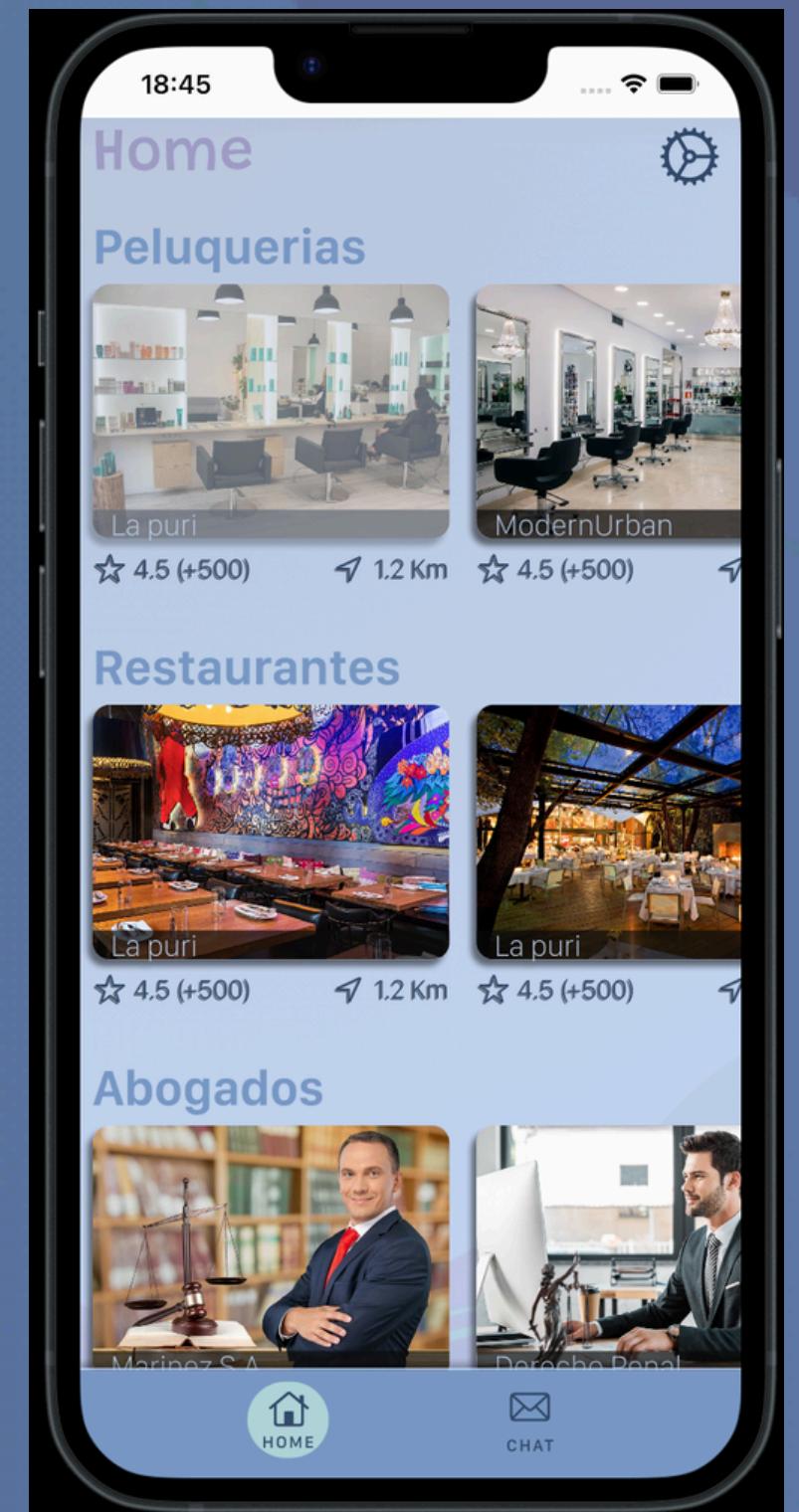
- Crecimiento del mercado gig
- Adopción global:
- Colaboraciones y asociaciones

¿Cómo?

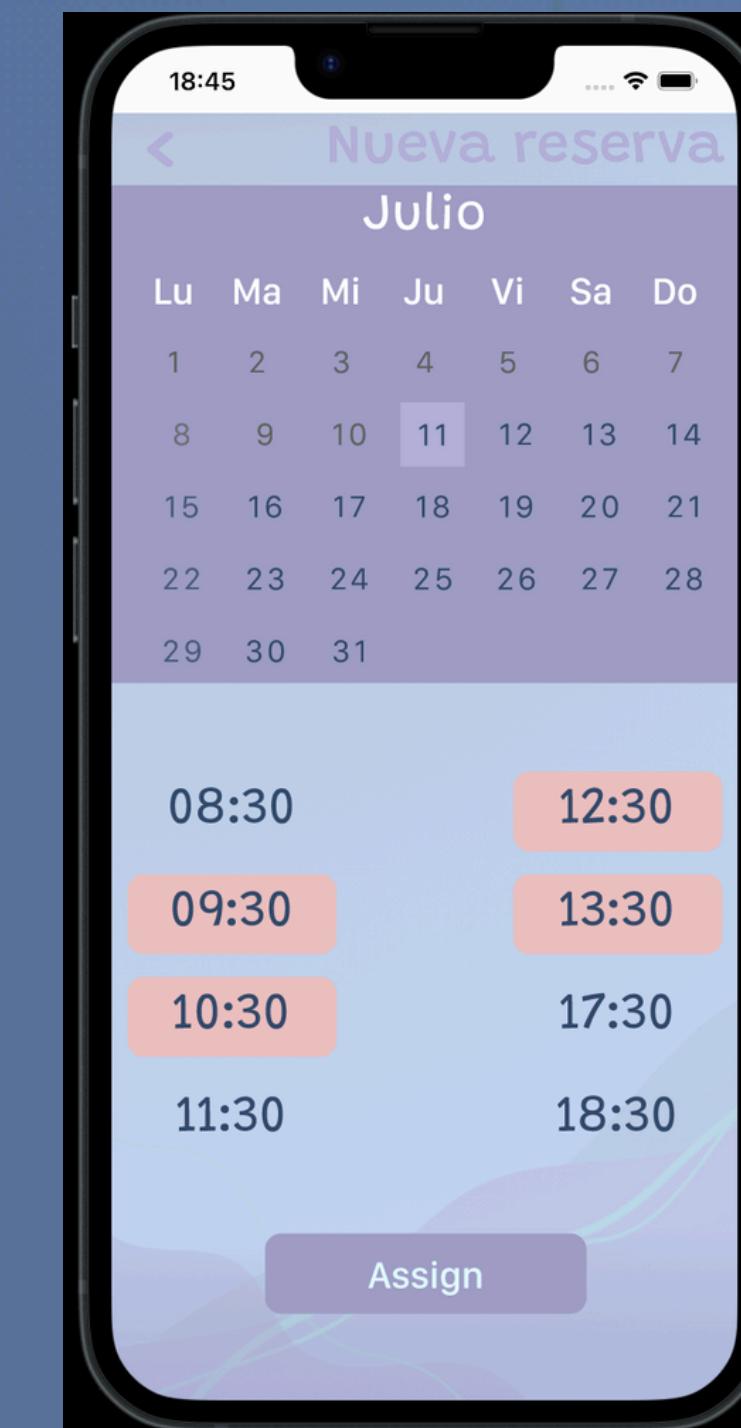
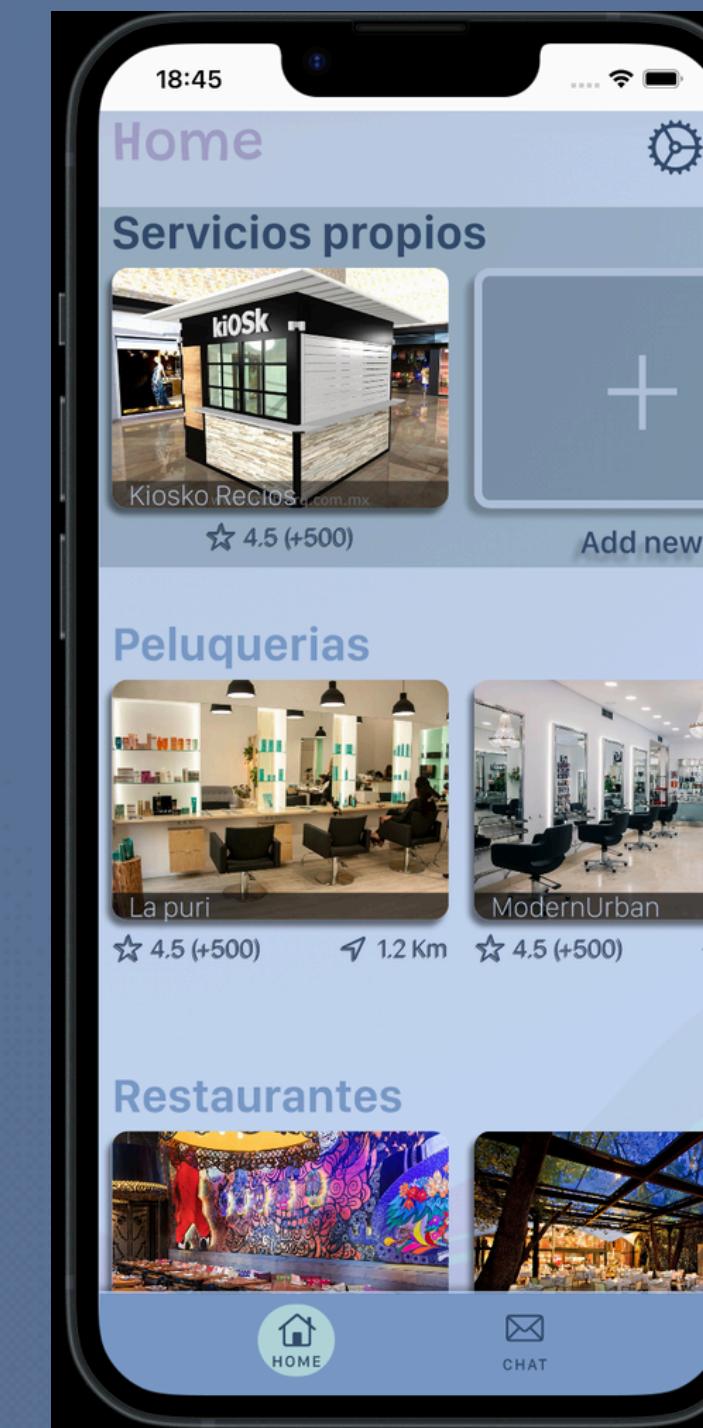
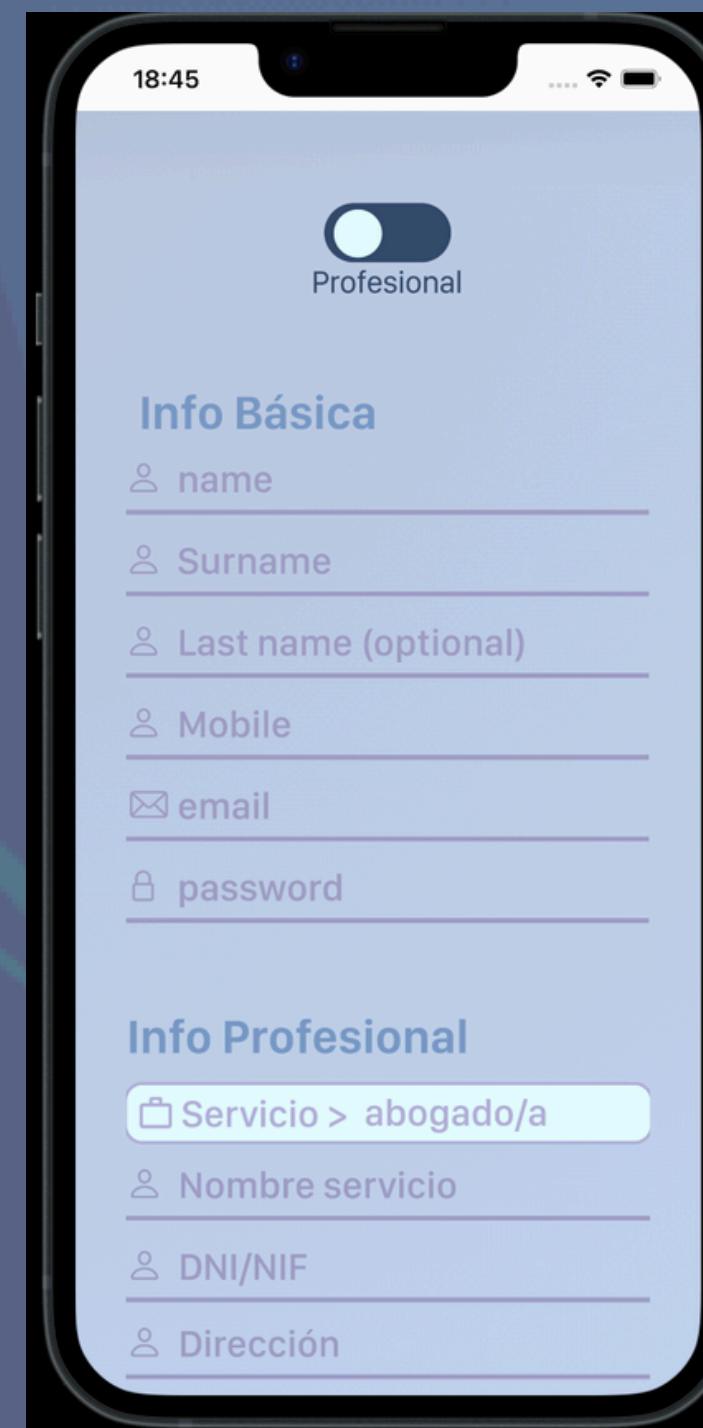
**ServiHub, app para promocionar y
contratar de servicios.**



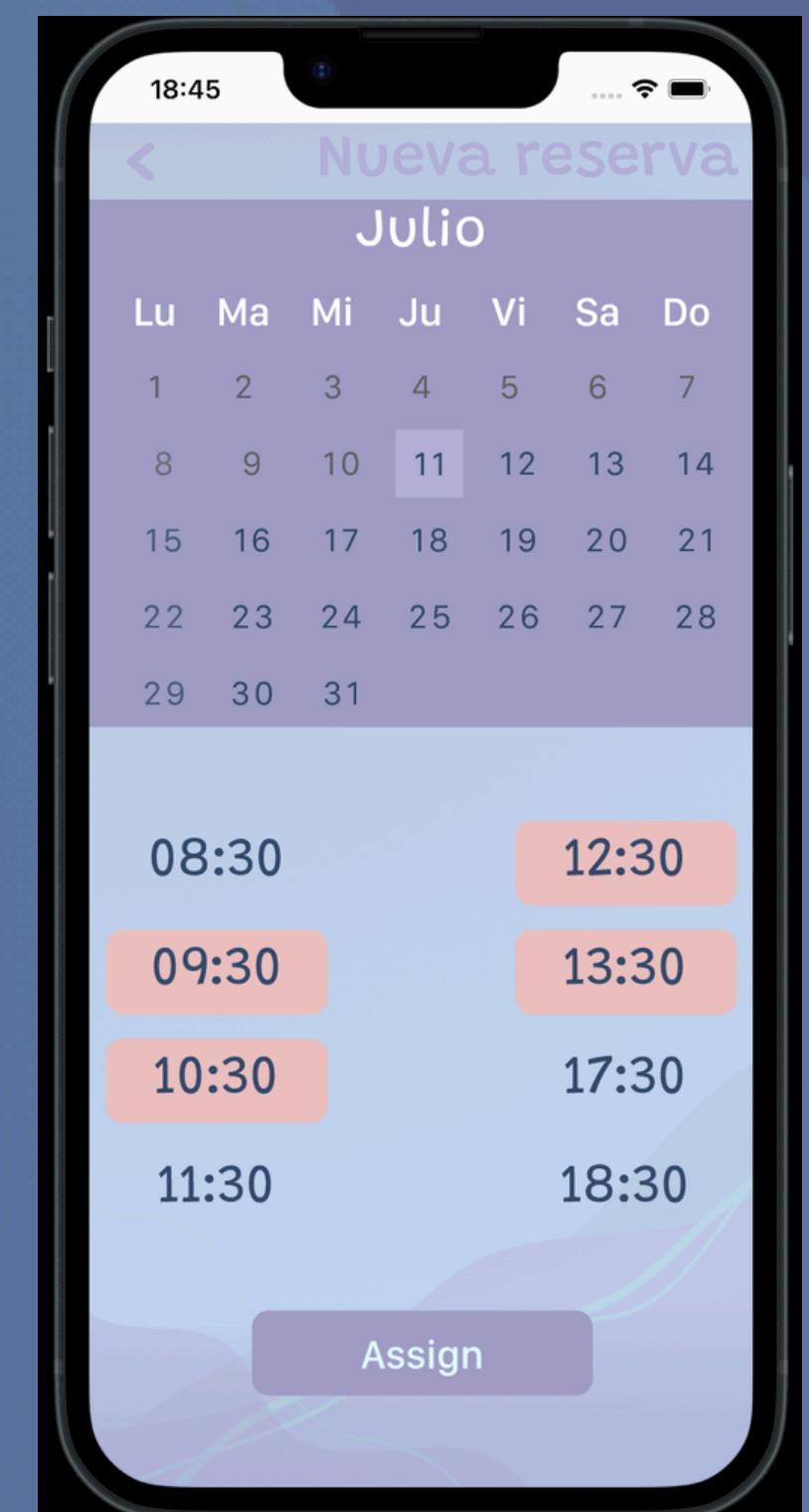
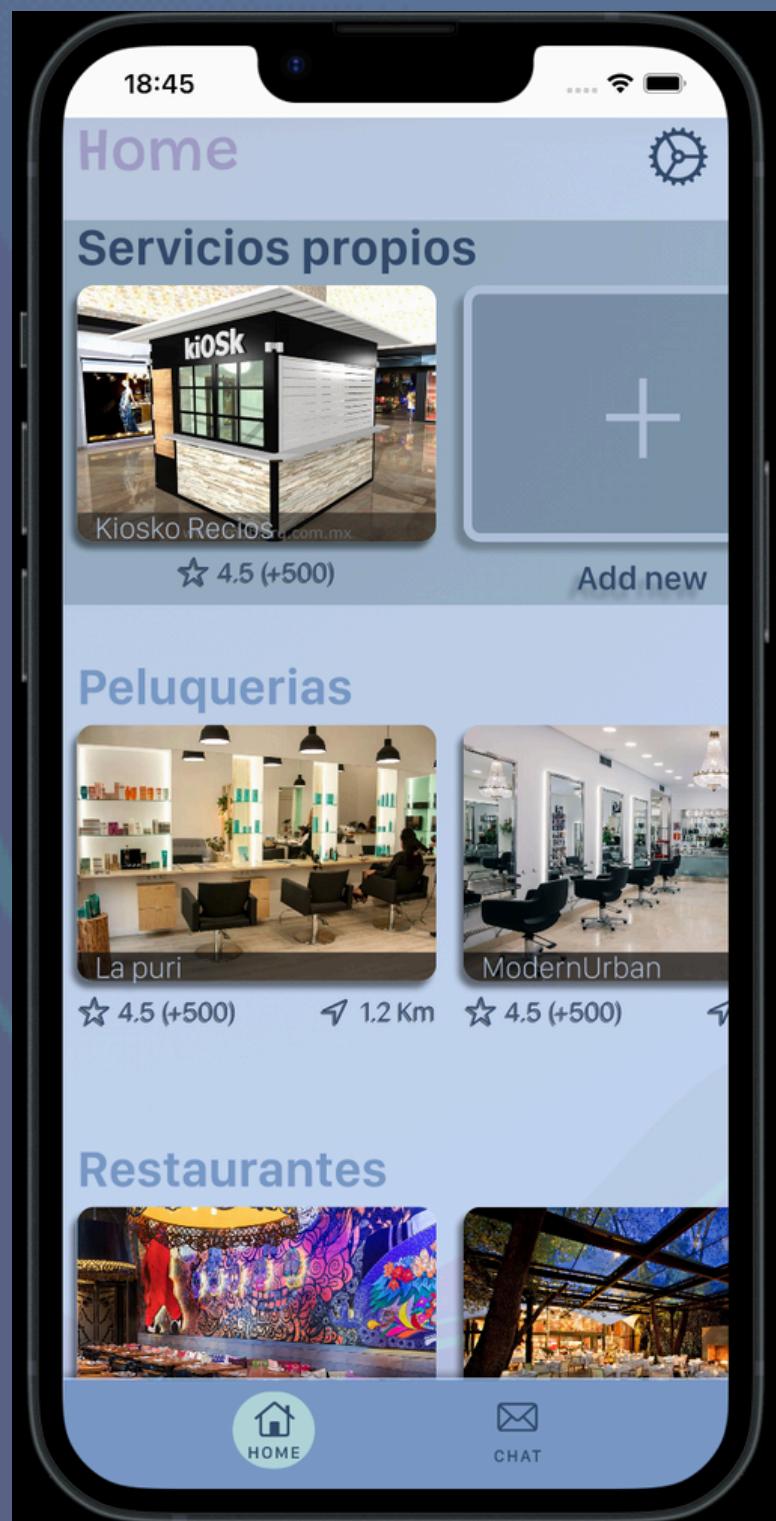
Con ServiHub queremos ayudar a los negocios locales y a la gente emprendedora a tener un ingreso a la vez que dan un trato perfecto al cliente.



Desde el momento que te registras como usuario o autónomo, podrás empezar a contratar servicios rápidamente.



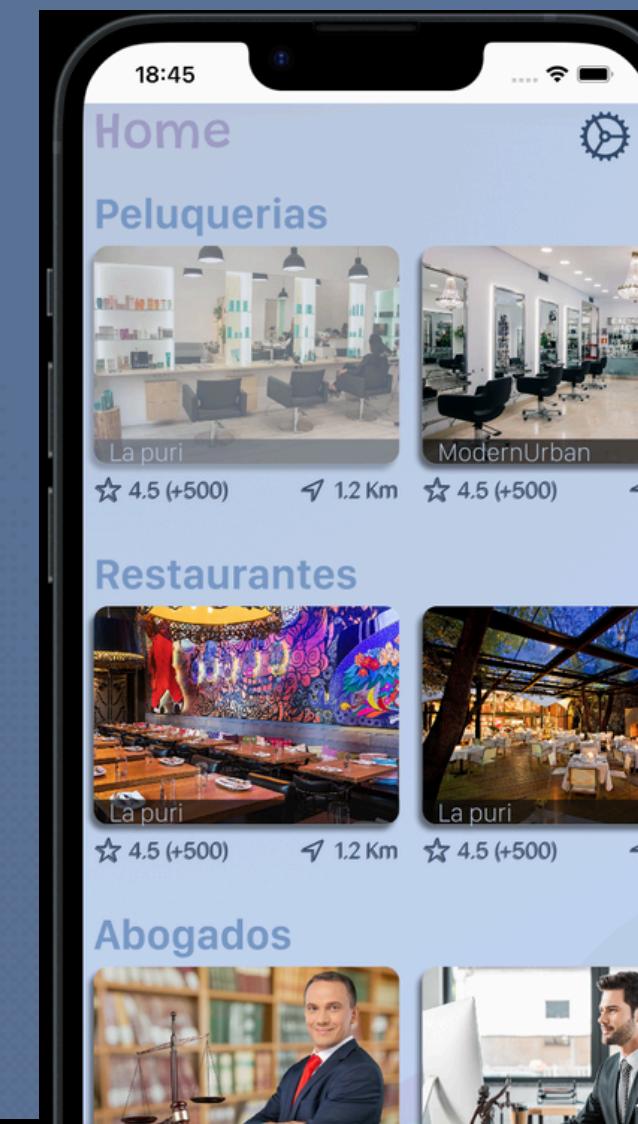
La aplicación te pondrá a disposición tu agenda para que los clientes tengan todas las facilidades para contratar tu servicio.



Esta app está pensada para los negocios locales que necesitan una agenda optimizada para aprovechar su tiempo de apertura al máximo.



El usuario podrá valorar y recomendar los servicios que haya contratado para que otros usuarios puedan ver cuáles son los mejores valorados y les sea más fácil elegir.



¿Quién(es)?

Los ultimos supervivientes

Start-up de desarrollo de aplicaciones multiplataforma para apple en Swift

- Swift 5
- SwiftUI
- MVVM
- Vapor
- Asyn/Await
- Keychain
- XCTest

Jose Bueno Cruz

Boncroix

Experto del desarrollo backend,
responsable del área de servidores e
implementación con las BBDD. Dice que
ahora sueña con tokens de autorización y su
personaje favorito es Marc Hervera

Diego Andrades

Diego 94

Experto y responsable del proyecto Servihub. Arquitectura, front end, domain, data... Tirale todo lo que quieras, que lo resuelve.

Cristian Contreras

Man at war

Todo un profesional de todos los ámbitos.
Igual te diseña, que te programa, que te
hace la cama y encima nunca tiene un pero.

Alejandro Gavira García

Man at war 2

Otro pesado multifunciones. Igual está a una cosa que te salta con otra. No se le puede tomar muy en serio.

Rocío Martos

Marketing y Asesoramiento legal

Profesional del ámbito del desarrollo económico y fiabilidad del proyecto. Dale derivadas que te las hace con las manos

Pablo Peragon

Marketing y Asesoramiento legal

Profesional del ambito del desarrollo económico y marco legal del proyecto. No sabe de leyes mucho, pero sus brazos son justicia y derecho.

¿Dónde?

El teletrabajo permite a los empleados organizar su horario laboral de acuerdo con sus necesidades personales, facilitando un mejor balance entre la vida laboral y personal.

El teletrabajo permite a las empresas contratar a los mejores talentos sin importar su ubicación geográfica.

Modelo de negocio.



Point Breaks

Se podrá realizar ofrecer servicios y gestionar reservas para los negocios. Tendrás un número limitado de reservas.

Ingreso por reserva

Comisión del 3% por el coste de la reserva.

Reef Breaks

Tendrá un coste fijo al més de 9,90€/més. El negocio podrá hacer ofertas y posicionar sus servicios en la app y tendrá un número ilimitado de reservas.

Posicionamiento

Las empresas podrán hacerse más visibles con esta opción.
100€/mes.

Soluciones SOS

Rondas de Inversión

Búsqueda de varias rondas de inversión

Ingresos por publicidad en la app

Comisión del 3% por el coste de la reserva.

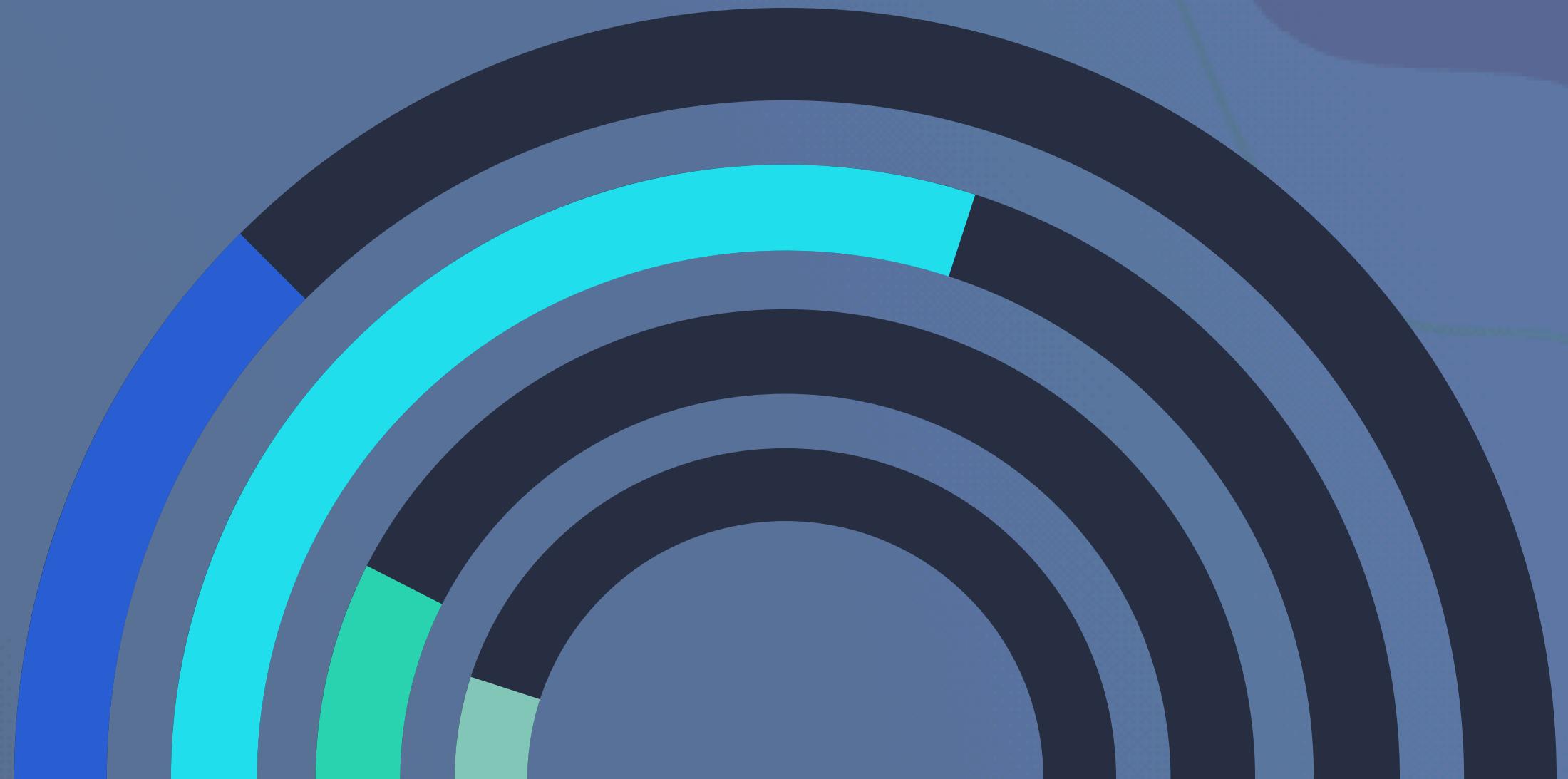
Subida de precios

Subida de comisiones y todos los precios una vez tengamos una base

Venta online

- 1. Estrategias de Venta Simplificadas**
- 2. Disparadores Mentales y Elementos de Apoyo**
- 3. Contenidos y Ofertas de Venta**
- 4. Webinarios y Embudos de Venta**

Planificación Financiera



UX



Desarrollo



Q/A



Marketing

Adquisición de usuarios

Lanzamiento y Campaña de conciencia: Utilizar campañas de marketing digital (SEM, SEO, redes sociales) para aumentar la visibilidad de la aplicación.

Colaboraciones y Promociones: Asociarse con proveedores populares para ofrecer promociones exclusivas a los usuarios.

Programas de Referidos: Implementar un programa de referidos para incentivar a los usuarios a invitar a otros.

Gastos proyecto



Gastos Sector



Salarios

500K

Marketing

50K

Desarrollo

250K

Anuncios Digitales

1. Anuncios por segmento demográfico
2. Redes sociales como enfoque principal
3. Contenidos patrocinados
4. AdSense y AdWorks
5. Webinars gratuitos

Influencers, expansiones de mercado, retención de usuarios, promociones estacionales

Planificación



Diseño de la plataforma

Preproducción: (0-3 meses)

- Prototipado UX/UI
- Estudio de mercado
- Dinamicas de grupo en el diseño
- Pruebas con versiones muy iniciales

Desarrollo de la plataforma

Producción: (3-12 meses)

- Hito 1: Alpha de la app
- Diseño completado
- Primeras pruebas con usuarios

Desarrollo de la plataforma

ProPost Producción: (12-18 meses)

- Feedback Cliente
- Revisión de la seguridad
- Testeo masivo de la app
- Salida por países

Desarrollo de la plataforma

Release: (18-24 meses)

- Versión 1.00.00
- Empieza el mantenimiento de la app

Arquitectura MVVM

Beneficios

1. Modularidad
2. Código reutilizable
3. Tests
4. Mocking sencillo con SwiftUI
5. Principios SOLID

Arquitectura MVVM

Desventajas

1. Inicio complicado
2. Realización de sobrecódigo
3. Trabajo por estados

Soluciones

1. Buena planificación
2. Trabajar con datos mockeados desde el inicio

Arquitectura MVVM

PRE PRODUCCIÓN

- 01** Estado inicial
Gestión y planificación
Fin UX/UI



PRODPOST

- 03** Primera entrega
Reevaluación
Inicio etapa postproducción
Seguridad



PRODUCCIÓN

- 02** Desarrollo de código
Backend



MANTENIMIENTO/ RELEASE

- 04** Inicio etapa release
Pruebas de estrategias de mercado
Marketing



Responsabilidad Social

Accesibilidad

La aplicación tendrá todo lo necesario para cumplir con los puntos de accesibilidad completo

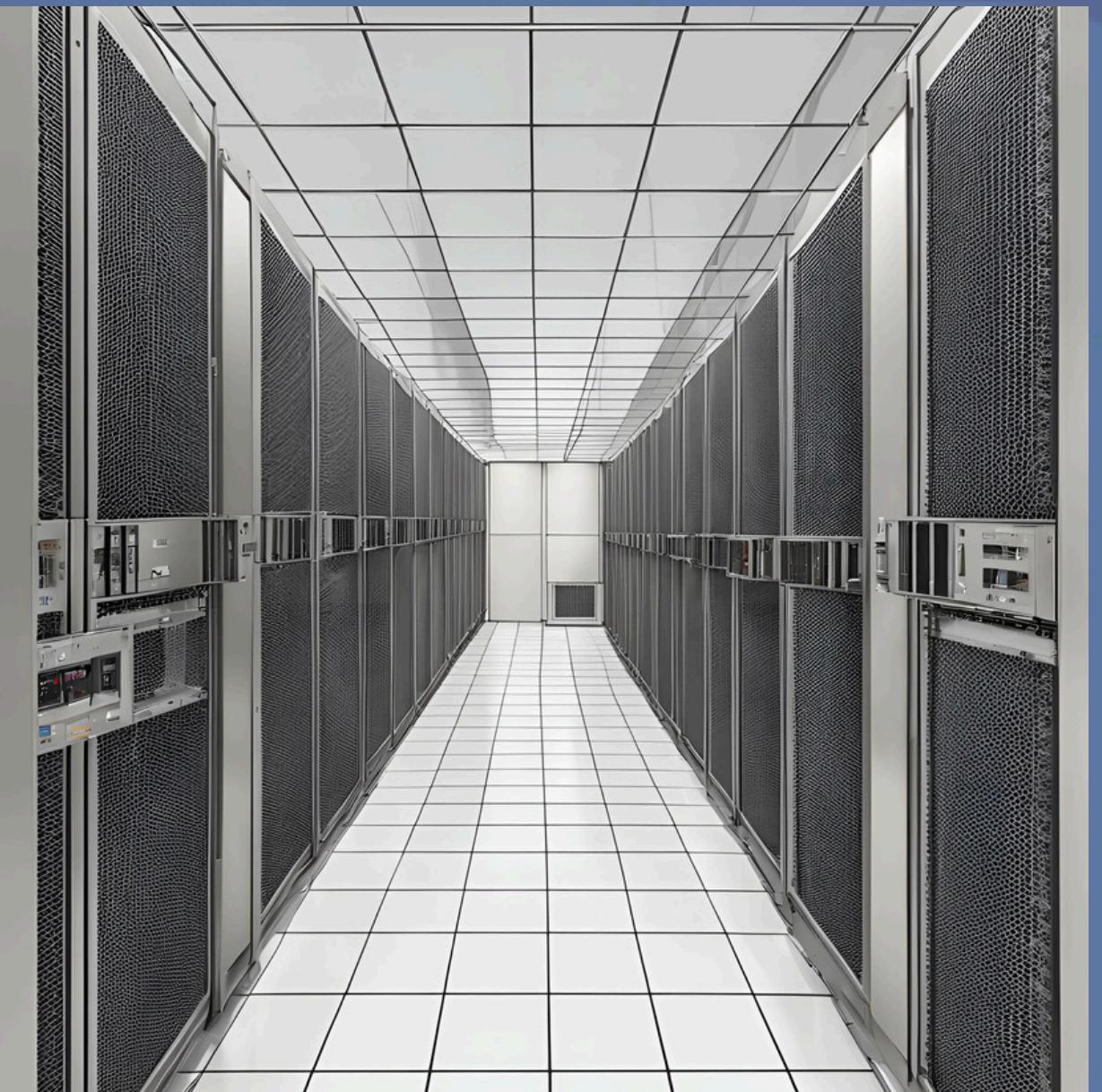
Apoyo

Prioridad de colaboración con negocios familiares o unipersonales. Apoyamos el crecimiento económico y su evolución

Reducción huella carbono

Compromiso con intentar el desarrollo sostenido y la máxima reducción posible para el futuro de todos nosotros

Server



Server - Servihub



Sistema de Backend para Conectar Clientes con Servicios Locales en ServiHub

Equipo de Desarrollo: Alejandro Alberto Garcia,
José Bueno, Cristian Contreras, Diego Andrades,
Pablo Peragón, Rocio Martos

Fecha: 02/08/2024

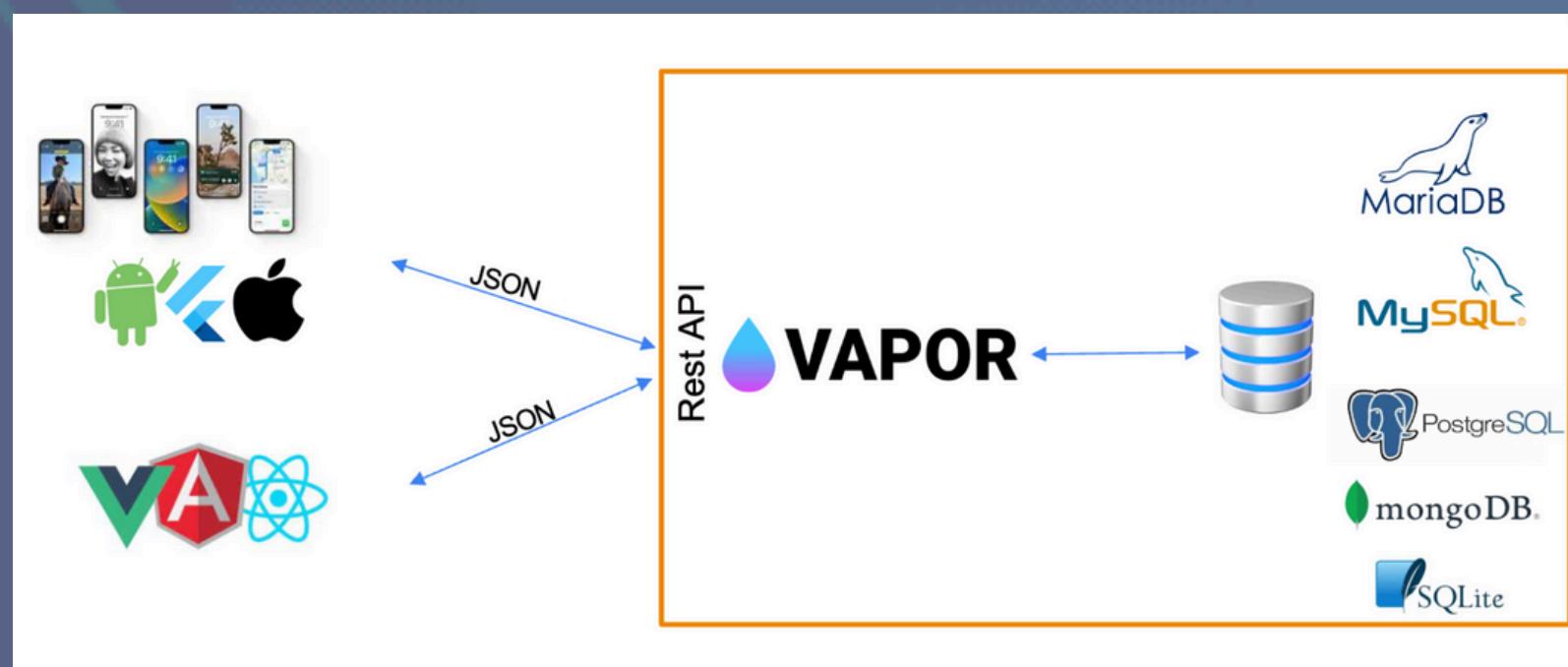
Índice:

1. Visión General del Servidor
2. Arquitectura del Servidor
3. Flujo de Datos
4. Beneficios y Objetivos
5. Público Objetivo
6. Consideraciones Técnicas
7. Casos de uso y Ejemplos
8. Seguridad y Privacidad
9. Mantenimiento y Soporte
10. Roadmap y Futuras Mejoras
11. Preguntas Frecuentes

1. Visión General del Servidor

- Descripción del Servidor: El servidor de ServiHub, desarrollado utilizando Xcode y Vapor, sirve como el backend de una aplicación que conecta a clientes con servicios locales. Permite el registro y gestión de dos tipos de usuarios: clientes y profesionales. Facilita la creación de servicios por parte de los profesionales y permite a los clientes buscar y reservar estos servicios.
- Propósito Principal: El propósito del servidor es manejar el registro y autenticación de usuarios, permitir la creación y gestión de servicios por parte de profesionales, y ofrecer a los clientes la capacidad de buscar, visualizar y reservar estos servicios.
- Características Clave:
 - Registro y autenticación de dos tipos de usuarios.
 - Creación y gestión de servicios por usuarios profesionales.
 - Búsqueda de servicios por ubicación, distancia, categoría y negocio.
 - Reservas directas y contacto con negocios desde la app.

2. Arquitectura del Servidor

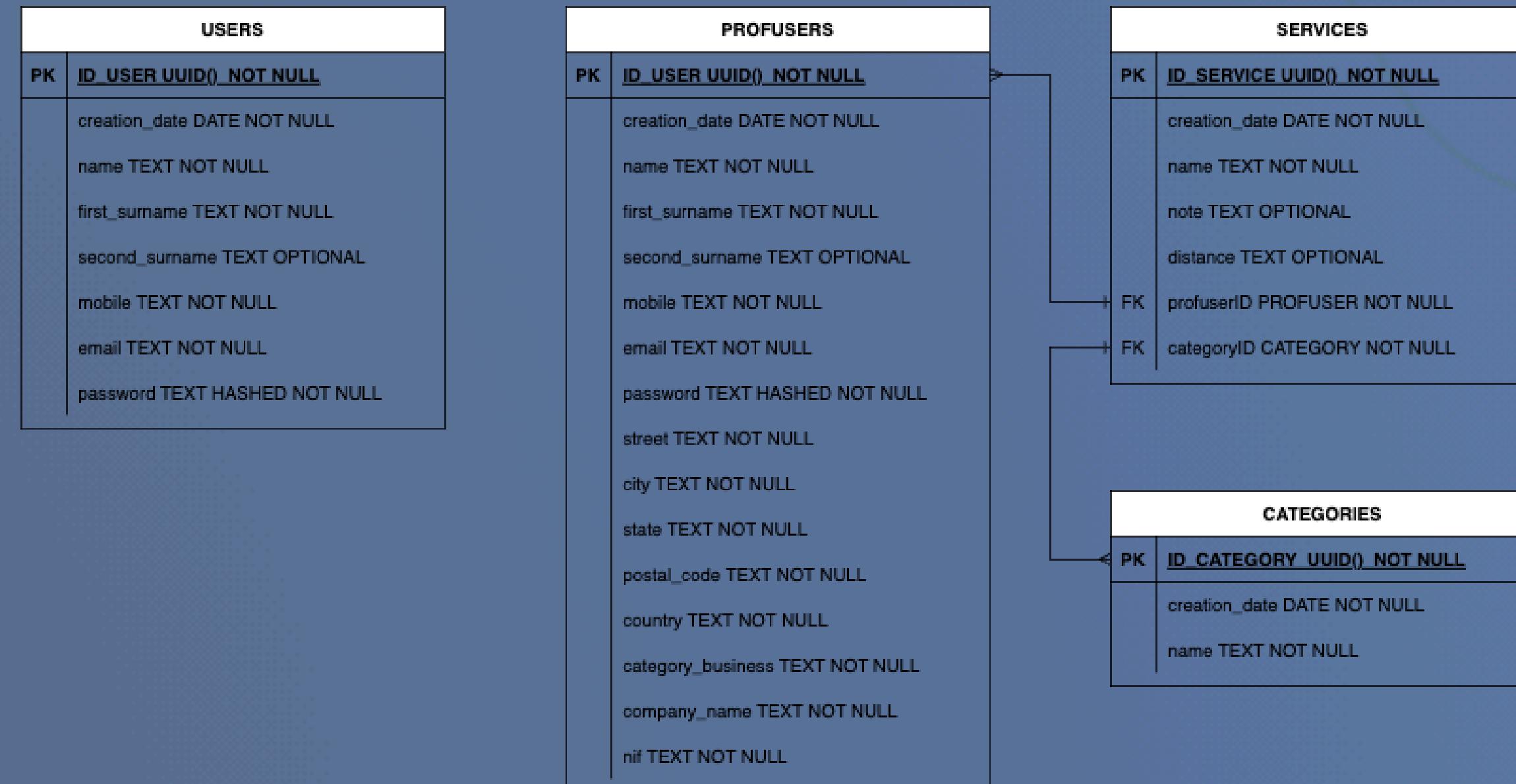


El servidor utiliza Vapor para el backend, que interactúa con una base de datos para almacenar la información de clientes y servicios en este caso utilizamos PostgreSQL. Las APIs RESTful permiten la comunicación con la app, facilitando operaciones como el registro de usuarios y la creación de servicios.

3. Flujo de Datos

Descripción del Flujo de Datos:

- Registro de Usuario:** El usuario se registra y proporciona datos específicos según su tipo (cliente o profesional).
- Creación de Servicios:** Los usuarios profesionales crean y gestionan sus servicios.
- Búsqueda y Reserva:** Los clientes buscan servicios, seleccionan y reservan o contactan a los negocios.



4. Beneficios y Objetivos

- **Beneficios del Servidor:**
 - **Para Profesionales:** Facilita la gestión y promoción de sus servicios.
 - **Para Clientes:** Proporciona una plataforma fácil de usar para encontrar y reservar servicios.
- **Objetivos de Diseño:**
 - **Eficiencia:** Procesar registros, búsquedas y reservas de manera rápida y eficiente.
 - **Escalabilidad:** Manejar un creciente número de usuarios y servicios sin comprometer el rendimiento.

Está diseñado para ofrecer una experiencia fluida y eficiente tanto para clientes como para profesionales. Su arquitectura modular asegura que pueda escalar fácilmente para manejar un aumento en la cantidad de usuarios y servicios.

5. Público Objeto

- **Usuarios del Servidor:**
 - **Clientes:** Personas que buscan y reservan servicios.
 - **Profesionales:** Negocios que crean y gestionan sus servicios.
- **Requisitos de Acceso:**
 - **Clientes:** Acceso para buscar y reservar servicios.
 - **Profesionales:** Acceso adicional para crear y gestionar servicios.

El servidor está diseñado para ser utilizado por dos tipos de usuarios: los clientes que buscan y reservan servicios, y los profesionales que crean y administran estos servicios.

6. Consideraciones Técnicas

- **Tecnologías Usadas:**
 - **Xcode:** Entorno de desarrollo para la creación de la app.
 - **Vapor:** Framework backend para la gestión de la lógica del servidor.
 - **Base de Datos:** Para almacenamiento de datos de usuarios y servicios.
- **Versiones y Compatibilidad:**
 - **Xcode:** Requiere al menos Xcode 11.0 o superior.
 - **Vapor 4:** Requiere Swift 5.1
 - **Base de Datos:** PostgreSQL

7. Casos de Uso y Ejemplos

Descripción General

Esta sección muestra cómo se utiliza el servidor en situaciones del mundo real. Incluye ejemplos prácticos de cómo los diferentes usuarios interactúan con el sistema.

Endpoints:

Versión	Auth	User Cliente	User Profesional	Categorías	Servicios
Get Version	Post Register Client	Get All Clients	Get All ProfUsers	Get All Categories	Get All Services
	Post Register ProfUser	Get Client By ID	Get ProfUser By ID	Get Category By ID	Get Service By ID
	Get Login Client	Put Update Client	Put Update ProfUser		Post Create Service
	Get Login ProfUser	Delete Client	Delete ProfUser		Put Update Service
	Get Refresh Client				Get All Services By ProfUser
	Get Refresh Tokens				Delete Service

Archivo creado en ApiRapi con todos los endpoints

<https://drive.google.com/file/d/1Kgz1EgjToYoNBF1VPAaoT0-MSMj9Ekn9/view?usp=sharing>

Endpoint: Verificación de Versión

Descripción: Este endpoint se utiliza para verificar la versión actual del servidor en comparación con la última versión disponible. Es útil para los clientes y desarrolladores que necesitan asegurarse de que están utilizando la versión más reciente del servidor.

Endpoint:

- URL: `http://127.0.0.1:8080/api/version`
- Método: GET

Parámetros de URL:

- **current (requerido):** Versión actual del cliente para comparar con la versión más reciente disponible.
- **Encabezados:**
- **SSH-ApiKey (requerido):** La API key de la aplicación para autenticar la solicitud.

Ejemplo de Solicitud:

http

GET http://127.0.0.1:8080/api/version?current=1.0.0

Headers:

SSH-ApiKey: your_api_key_here

Ejemplo de Respuesta:

json

```
{  
  "needsUpdate": true,  
  "current": "1.0.0",  
  "live": "1.0.8"  
}
```

Descripción de la Respuesta:

- **needsUpdate:** Indica si el cliente necesita actualizarse (true si es necesario, false si no).
- **current:** Versión actual del cliente que hizo la solicitud.
- **live:** Última versión disponible del servidor.

Casos de Uso:

- **Verificación de Actualización de Versión:**
 - Escenario: Un cliente o desarrollador necesita confirmar si su aplicación está actualizada.
 - Proceso:
 - El cliente envía una solicitud GET al endpoint /api/version, incluyendo su versión actual en el parámetro current y el encabezado SSH-ApiKey con la clave de API (your_api_key_here).
 - El servidor responde con la versión actual del cliente y la última versión disponible.
 - El cliente verifica el valor de needsUpdate para determinar si debe proceder con la actualización.
- **Actualización de Aplicaciones:**
 - Escenario: Un sistema automatizado de actualización necesita comprobar si hay una nueva versión disponible antes de realizar un despliegue.
 - Proceso:
 - El sistema envía una solicitud GET al endpoint /api/version con la versión actual (1.0.0) y el encabezado SSH-ApiKey con la clave de API.
 - Analiza la respuesta para ver si needsUpdate es true.
 - Si es necesario, el sistema inicia la actualización del cliente a la última versión.

Endpoint: Registro

Descripción: Este endpoint permite a un nuevo cliente registrarse en la plataforma. Los datos proporcionados se utilizan para crear una nueva cuenta en el sistema.

Endpoint:

- Cliente: <http://127.0.0.1:8080/api/auth/signup>
- Usuario Profesional: <http://127.0.0.1:8080/api/auth/profsignup>
- Método: POST

Encabezados:

- **SSH-ApiKey (requerido):** La API key de la aplicación para autenticar la solicitud.

Ejemplo de Solicitud:

http Client

```
POST http://127.0.0.1:8080/api/auth/signup
```

Headers:

SSH-ApiKey: your_api_key_here

Body:

```
{  
  "name": "John",  
  "firstSurname": "Doe",  
  "secondSurname": "Smith",  
  "mobile": "123456789",  
  "email": "john.doe@example.com",  
  "password": "securepassword123"  
}
```

http ProfUser

```
POST http://127.0.0.1:8080/api/auth/signup
```

Headers:

SSH-ApiKey: your_api_key_here

Body:

```
{  
  "name": "María",  
  "firstSurname": "García",  
  "secondSurname": "Fernández",  
  "mobile": "600987654",  
  "email": "maria.garcia@example.com",  
  "password": "securepassword123",  
  "street": "Avenida de la Paz",  
  "city": "Madrid",  
  "state": "Madrid",  
  "postalCode": "28001",  
  "country": "España",  
  "categoryBusiness": "estetica",  
  "companyName": "Estética Belleza",  
  "nif": "A12345678"  
}
```

Ejemplo de Respuesta:

json

```
{  
  "userID": "your user ID",  
  "accessToken": "your accessToken",  
  "refreshToken": "your refreshToken"  
}
```

Descripción de la Respuesta:

- **userID:** Identificador único del nuevo cliente creado.
- **accessToken:** Token de acceso generado para el cliente.
- **refreshToken:** Token de refresco para obtener nuevos tokens de acceso.

Casos de Uso:

- **Registro de Nuevo Cliente:**
 - Escenario: Un nuevo usuario desea crear una cuenta en la plataforma.
 - Proceso:
 - El usuario envía una solicitud POST al endpoint /auth/signup, incluyendo el encabezado SSH-ApiKey con la clave de API y el cuerpo con la información requerida (name, firstSurname, mobile, email, password) y opcionalmente secondSurname.
 - El servidor procesa la solicitud y crea una nueva cuenta.
 - El servidor responde con los identificadores y tokens necesarios para el nuevo cliente.
- **Integración en Aplicaciones:**
 - Escenario: Una aplicación móvil necesita permitir a los usuarios registrarse en la plataforma.
 - Proceso:
 - La aplicación móvil envía una solicitud POST al endpoint <http://127.0.0.1:8080/api/auth/signup> con los datos del cliente y el encabezado SSH-ApiKey.
 - La aplicación maneja la respuesta para almacenar los tokens y el identificador del usuario para futuras autenticaciones.

Endpoints: Inicio de Sesión

Descripción: Este endpoint permite a un cliente iniciar sesión en la plataforma. Los datos proporcionados se utilizan para autenticar al cliente y generar tokens de acceso.

Endpoint:

- Cliente: <http://127.0.0.1:8080/api/auth/signin>
- Usuario Profesional: <http://127.0.0.1:8080/api/auth/profsignin>
- Método: GET

Encabezados:

- **SSH-ApiKey (requerido):** La API key de la aplicación para autenticar la solicitud.
- **Authorization (requerido):** El valor debe ser Basic seguido de las credenciales del cliente (email:password) codificadas en Base64.

Ejemplo de Solicitud:

http Client and ProfUser

GET http://127.0.0.1:8080/api/auth/signin

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Basic am9obi5kb2VAZXhhbXBsZS5jb206c2VjdXJlcGFzc3dvcvmQxMjM=

Ejemplo de Valor para el Encabezado Authorization:

- Credenciales: john.doe@example.com:securepassword123
- Codificado en Base64: am9obi5kb2VAZXhhbXBsZS5jb206c2VjdXJlcGFzc3dvcvmQxMjM=

GET http://127.0.0.1:8080/api/auth/profsignin

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Basic am9obi5kb2VAZXhhbXBsZS5jb206c2VjdXJlcGFzc3dvcvmQxMjM=

Ejemplo de Respuesta:

json

```
{  
  "userID": "your user ID",  
  "accessToken": "your accessToken",  
  "refreshToken": "your refreshToken"  
}
```

Descripción de la Respuesta:

- **userID:** Identificador único del nuevo cliente creado.
- **accessToken:** Token de acceso generado para el cliente.
- **refreshToken:** Token de refresco para obtener nuevos tokens de acceso.

Casos de Uso:

- **Inicio de Sesión de Cliente:**
 - Escenario: Un cliente desea iniciar sesión en la plataforma.
 - Proceso:
 - El cliente envía una solicitud GET al endpoint /api/auth/signin, incluyendo los encabezados SSH-ApiKey y Authorization con las credenciales codificadas en Base64.
 - El servidor procesa la solicitud y autentica al cliente.
 - El servidor responde con los identificadores y tokens necesarios para el cliente autenticado.
- **Integración en Aplicaciones:**
 - Escenario: Una aplicación móvil necesita permitir a los usuarios iniciar sesión en la plataforma.
 - Proceso:
 - La aplicación móvil envía una solicitud GET al endpoint http://127.0.0.1:8080/api/auth/signin con los encabezados SSH-ApiKey y Authorization.
 - La aplicación maneja la respuesta para almacenar los tokens y el identificador del usuario para futuras autenticaciones.

Endpoints: Refresh User Tokens

Descripción: Este endpoint permite refrescar los tokens de acceso de un usuario (cliente o usuario profesional) utilizando el correo electrónico y la contraseña. Los datos proporcionados varían según el tipo de usuario.

Endpoints:

- Cliente: <http://127.0.0.1:8080/api/auth/refresh>
- Usuario Profesional: <http://127.0.0.1:8080/api/auth/prorefresh>
- Método: GET

Encabezados:

- SSH-ApiKey (requerido): La API key de la aplicación para autenticar la solicitud.
- Authorization (requerido): El valor debe ser Bearer seguido del refresh token.

Ejemplo de Solicitud:

http Client and ProfUser

GET http://127.0.0.1:8080/api/auth/refresh

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

GET http://127.0.0.1:8080/api/auth/prorefresh

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

Ejemplo de Valor para el Encabezado Authorization:

- Refresh Token: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

Ejemplo de Respuesta:

json

```
{  
  "userID": "your user ID",  
  "accessToken": "your accessToken",  
  "refreshToken": "your refreshToken"  
}
```

Descripción de la Respuesta:

- **userID:** Identificador único del nuevo cliente creado.
- **accessToken:** Token de acceso generado para el cliente.
- **refreshToken:** Token de refresco para obtener nuevos tokens de acceso.

Casos de Uso:

- **Refresh Token de Cliente:**
 - Escenario: Un cliente desea refreshar su token de acceso.
 - Proceso:
 - El cliente envía una solicitud GET al endpoint /api/auth/frefresh, incluyendo los encabezados SSH-ApiKey y Authorization con las credenciales.
 - El servidor procesa la solicitud y autentica al cliente.
 - El servidor responde con los nuevos identificadores y tokens necesarios para el cliente autenticado.
- **Refresh Token de Usuario Profesional:**
 - Escenario: Un usuario profesional desea refreshar su token de acceso.
 - Proceso:
 - El usuario profesional envía una solicitud GET al endpoint /api/auth/profrefresh, incluyendo los encabezados SSH-ApiKey y Authorization con las credenciales.
 - El servidor procesa la solicitud y autentica al usuario profesional.
 - El servidor responde con los nuevos identificadores y tokens necesarios para el usuario profesional autenticado.

Endpoints: Obtener todos los Usuarios

Descripción: Este endpoint permite obtener una lista de todos los usuarios registrados en el sistema, tanto clientes como usuarios profesionales. La respuesta varía según el tipo de usuario solicitado.

Endpoints:

- Clientes: <http://127.0.0.1:8080/api/users/getallusers>
- Usuarios Profesionales: <http://127.0.0.1:8080/api/users/getallprofusers>
- Método: GET

Encabezados:

- SSH-ApiKey (requerido): La API key de la aplicación para autenticar la solicitud.
- Authorization (requerido): El valor debe ser Bearer seguido del refresh token.

Ejemplo de Solicitud:

http Client and ProfUser

GET http://127.0.0.1:8080/api/users/getallusers

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

Ejemplo de Valor para el Encabezado Authorization:

- Refresh Token: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

GET http://127.0.0.1:8080/api/users/getallprofusers

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

Ejemplo de Respuesta:

json Client List

```
[  
  {  
    "id": "lkLJIJSIFLEJKLJ54876S4FE54S",  
    "name": "John",  
    "firstSurname": "Doe",  
    "secondSurname": "Smith",  
    "mobile": "123456789",  
    "email": "john.doe@example.com",  
  }  
]
```

- **Clientes:**

- **id:** Identificador único del cliente.
- **name:** Nombre del cliente.
- **firstSurname:** Primer apellido del cliente.
- **secondSurname:** Segundo apellido del cliente (opcional).
- **mobile:** Número de teléfono móvil del cliente.
- **email:** Dirección de correo electrónico del cliente.

json ProfUser List

```
[  
  {  
    "id": "lkLJIJSIFLEJKLJ54876S4FE54S",  
    "name": "María",  
    "firstSurname": "García",  
    "secondSurname": "Fernández",  
    "mobile": "600987654",  
    "email": "maria.garcia@example.com",  
    "street": "Avenida de la Paz",  
    "city": "Madrid",  
    "state": "Madrid",  
    "postalCode": "28001",  
    "country": "España",  
    "categoryBusiness": "estetica",  
    "companyName": "Estética Belleza",  
    "nif": "A12345678"  
  }  
]
```

Descripción de la Respuesta:

- **Usuarios Profesionales:**
 - **id:** Identificador único del usuario profesional.
 - **name:** Nombre del usuario profesional.
 - **firstSurname:** Primer apellido del usuario profesional.
 - **secondSurname:** Segundo apellido del usuario profesional (opcional).
 - **mobile:** Número de teléfono móvil del usuario profesional.
 - **email:** Dirección de correo electrónico del usuario profesional.
 - **street:** Calle de la dirección del negocio.
 - **city:** Ciudad de la dirección del negocio.
 - **state:** Estado de la dirección del negocio.
 - **postalCode:** Código postal de la dirección del negocio.
 - **country:** País de la dirección del negocio.
 - **categoryBusiness:** Categoría del negocio.
 - **companyName:** Nombre de la empresa.
 - **nif:** NIF del negocio.

Casos de Uso:

- Obtener Todos los Clientes:
 - Escenario: Un administrador desea obtener una lista de todos los clientes registrados en el sistema.
 - Proceso:
 - El administrador envía una solicitud GET al endpoint /api/users/getallusers, incluyendo los encabezados SSH-ApiKey y Authorization con el refresh token.
 - El servidor procesa la solicitud y autentica al administrador.
 - El servidor responde con una lista de todos los clientes registrados.
- Obtener Todos los Usuarios Profesionales:
 - Escenario: Un administrador desea obtener una lista de todos los usuarios profesionales registrados en el sistema.
 - Proceso:
 - El administrador envía una solicitud GET al endpoint /api/users/getallprofusers, incluyendo los encabezados SSH-ApiKey y Authorization con el refresh token.
 - El servidor procesa la solicitud y autentica al administrador.
 - El servidor responde con una lista de todos los usuarios profesionales registrados.

Endpoints: Obtener Usuario por ID

Descripción:

Este endpoint permite obtener la información de un usuario específico, ya sea un cliente o un usuario profesional, mediante su ID.

Endpoints:

- Cliente: <http://127.0.0.1:8080/api/users/geuser/{id}>
- Usuario Profesional: <http://127.0.0.1:8080/api/users/getprofuser/{id}>
- Método: GET

Parámetros de URL:

- {id}: El ID del usuario que se desea obtener.

Encabezados:

- SSH-ApiKey (requerido): La API key de la aplicación para autenticar la solicitud.
- Authorization (requerido): El valor debe ser Bearer seguido del refresh token.

Ejemplo de Valor para el Encabezado Authorization:

- Refresh Token: Bearer
`eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...`

Ejemplo de Solicitud:

http Client and ProfUser

GET http://127.0.0.1:8080/api/users/getuser/{idUser}

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

GET http://127.0.0.1:8080/api/users/getprofuser/{idUser}

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

Ejemplo de Respuesta:

json Client

```
{  
  "id": "lkLJIJSIFLEJKLJ54876S4FE54S",  
  "name": "John",  
  "firstSurname": "Doe",  
  "secondSurname": "Smith",  
  "mobile": "123456789",  
  "email": "john.doe@example.com",  
}
```

Descripción de la Respuesta:

- **Clientes:**
 - **id:** Identificador único del cliente.
 - **name:** Nombre del cliente.
 - **firstSurname:** Primer apellido del cliente.
 - **secondSurname:** Segundo apellido del cliente (opcional).
 - **mobile:** Número de teléfono móvil del cliente.
 - **email:** Dirección de correo electrónico del cliente.

json ProfUser

```
{  
  "id": "lkLJIJSIFLEJKLJ54876S4FE54S",  
  "name": "María",  
  "firstSurname": "García",  
  "secondSurname": "Fernández",  
  "mobile": "600987654",  
  "email": "maria.garcia@example.com",  
  "street": "Avenida de la Paz",  
  "city": "Madrid",  
  "state": "Madrid",  
  "postalCode": "28001",  
  "country": "España",  
  "categoryBusiness": "estetica",  
  "companyName": "Estética Belleza",  
  "nif": "A12345678"  
}
```

Descripción de la Respuesta:

- **Usuarios Profesionales:**
 - **id:** Identificador único del usuario profesional.
 - **name:** Nombre del usuario profesional.
 - **firstSurname:** Primer apellido del usuario profesional.
 - **secondSurname:** Segundo apellido del usuario profesional (opcional).
 - **mobile:** Número de teléfono móvil del usuario profesional.
 - **email:** Dirección de correo electrónico del usuario profesional.
 - **street:** Calle de la dirección del negocio.
 - **city:** Ciudad de la dirección del negocio.
 - **state:** Estado de la dirección del negocio.
 - **postalCode:** Código postal de la dirección del negocio.
 - **country:** País de la dirección del negocio.
 - **categoryBusiness:** Categoría del negocio.
 - **companyName:** Nombre de la empresa.
 - **nif:** NIF del negocio.

Casos de Uso:

- Obtener Información de un Cliente por ID:
 - Escenario: Un administrador desea obtener la información específica de un cliente.
 - Proceso:
 - El administrador envía una solicitud GET al endpoint `/api/users/getuser/{id}`, incluyendo los encabezados `SSH-ApiKey` y `Authorization` con el refresh token.
 - El servidor procesa la solicitud y autentica al administrador.
 - El servidor responde con la información del cliente solicitado.
- Obtener Información de un Usuario Profesional por ID:
 - Escenario: Un administrador desea obtener la información específica de un usuario profesional.
 - Proceso:
 - El administrador envía una solicitud GET al endpoint `/api/users/getprofuser/{id}`, incluyendo los encabezados `SSH-ApiKey` y `Authorization` con el refresh token.
 - El servidor procesa la solicitud y autentica al administrador.
 - El servidor responde con la información del usuario profesional solicitado.

Integración en Aplicaciones:

Escenario: Una aplicación móvil necesita permitir a los administradores obtener información específica de un usuario registrado en el sistema, ya sea un cliente o un usuario profesional.

Endpoints: Actualizar Usuarios

Descripción: Estos endpoints permiten actualizar la información de un usuario específico, ya sea un cliente o un usuario profesional, mediante su ID.

Endpoint:

- Cliente: <http://127.0.0.1:8080/api/users/updateuser/{idUser}>
- Usuario Profesional: <http://127.0.0.1:8080/api/users/updateprofuser/{idUser}>
- Método: PUT

Encabezados:

- SSH-ApiKey (requerido): La API key de la aplicación para autenticar la solicitud.
- Authorization (requerido): El valor debe ser Bearer seguido del refresh token.

Ejemplo de Solicitud:

http Client

```
PUT http://127.0.0.1:8080/api/users/updateprofuser/{idUser}
```

Headers:

SSH-ApiKey: your_api_key_here

Body:

```
{  
  "name": "John",  
  "firstSurname": "Doe",  
  "secondSurname": "Smith",  
  "mobile": "123456789",  
  "email": "john.doe@example.com",  
  "password": "securepassword123"  
}
```

http ProfUser

```
PUT http://127.0.0.1:8080/api/users/updateprofuser/{idUser}
```

Headers:

SSH-ApiKey: your_api_key_here

Body:

```
{  
  "name": "María",  
  "firstSurname": "García",  
  "secondSurname": "Fernández",  
  "mobile": "600987654",  
  "email": "maria.garcia@example.com",  
  "password": "securepassword123",  
  "street": "Avenida de la Paz",  
  "city": "Madrid",  
  "state": "Madrid",  
  "postalCode": "28001",  
  "country": "España",  
  "categoryBusiness": "estetica",  
  "companyName": "Estética Belleza",  
  "nif": "A12345678"  
}
```

Ejemplo de Respuesta:

json Client

```
{  
  "id": "lkLJIJSIFLEJKLJ54876S4FE54S",  
  "name": "John",  
  "firstSurname": "Doe",  
  "secondSurname": "Smith",  
  "mobile": "123456789",  
  "email": "john.doe@example.com",  
}
```

Descripción de la Respuesta:

- **Clientes:**
 - **id:** Identificador único del cliente.
 - **name:** Nombre del cliente.
 - **firstSurname:** Primer apellido del cliente.
 - **secondSurname:** Segundo apellido del cliente (opcional).
 - **mobile:** Número de teléfono móvil del cliente.
 - **email:** Dirección de correo electrónico del cliente.

json ProfUser

```
{  
  "id": "lkLJIJSIFLEJKLJ54876S4FE54S",  
  "name": "María",  
  "firstSurname": "García",  
  "secondSurname": "Fernández",  
  "mobile": "600987654",  
  "email": "maria.garcia@example.com",  
  "street": "Avenida de la Paz",  
  "city": "Madrid",  
  "state": "Madrid",  
  "postalCode": "28001",  
  "country": "España",  
  "categoryBusiness": "estetica",  
  "companyName": "Estética Belleza",  
  "nif": "A12345678"  
}
```

Descripción de la Respuesta:

- **Usuarios Profesionales:**
 - **id:** Identificador único del usuario profesional.
 - **name:** Nombre del usuario profesional.
 - **firstSurname:** Primer apellido del usuario profesional.
 - **secondSurname:** Segundo apellido del usuario profesional (opcional).
 - **mobile:** Número de teléfono móvil del usuario profesional.
 - **email:** Dirección de correo electrónico del usuario profesional.
 - **street:** Calle de la dirección del negocio.
 - **city:** Ciudad de la dirección del negocio.
 - **state:** Estado de la dirección del negocio.
 - **postalCode:** Código postal de la dirección del negocio.
 - **country:** País de la dirección del negocio.
 - **categoryBusiness:** Categoría del negocio.
 - **companyName:** Nombre de la empresa.
 - **nif:** NIF del negocio.

Casos de Uso:

- Actualizar Información de un Cliente:
 - Escenario: Un cliente desea actualizar su información de contacto.
 - Proceso:
 - El cliente envía una solicitud PUT al endpoint `/api/users/updateuser/{idUser}`, incluyendo los encabezados `SSH-ApiKey` y `Authorization` con el refresh token.
 - El servidor procesa la solicitud y actualiza la información del cliente.
 - El servidor responde con la información actualizada del cliente.
- Actualizar Información de un Usuario Profesional:
 - Escenario: Un usuario profesional desea actualizar su información de negocio.
 - Proceso:
 - El usuario profesional envía una solicitud PUT al endpoint `/api/users/updateprofuser/{idUser}`, incluyendo los encabezados `SSH-ApiKey` y `Authorization` con el refresh token.
 - El servidor procesa la solicitud y actualiza la información del usuario profesional.
 - El servidor responde con la información actualizada del usuario profesional.

Integración en Aplicaciones:

Escenario: Una aplicación móvil necesita permitir a los usuarios actualizar su información registrada en la plataforma.

Endpoints: Delete User

Descripción

Estos endpoints permiten eliminar usuarios de la base de datos, tanto para usuarios normales como para usuarios profesionales.

Endpoint 1: Borrar Usuario

- Método HTTP: DELETE
- Cliente: <http://127.0.0.1:8080/api/users/deleteuser/{idUser}>
- Usuario Profesional: <http://127.0.0.1:8080/api/users/deleteprofuser/{idUser}>

Headers

- SSH-ApiKey: {apiKey}
- Authorization: Bearer {refreshToken}

Parámetros

- Path Parameter:
 - idUser (string) - El ID del usuario que se desea eliminar.

Respuesta

- Código HTTP: 204 No Content

Ejemplo de Valor para el Encabezado Authorization:

- Refresh Token: Bearer
`eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...`

Ejemplo de Solicitud:

http Client and ProfUser

DELETE http://127.0.0.1:8080/api/users/deleteuser/{userID}

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

DELETE http://127.0.0.1:8080/api/users/deleteprofuser{userID}

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWliOiI5OEE3Q...

Proceso en el Servidor

La función en el servidor busca al usuario por su ID y, si lo encuentra, lo elimina de la base de datos.

La respuesta HTTP es 204 No Content, indicando que la operación se realizó exitosamente y no hay contenido adicional que devolver.

Casos de Uso

Escenario: Eliminación de Usuario

Un administrador desea eliminar un usuario específico de la base de datos.

Proceso

1. El administrador envía una solicitud DELETE al endpoint correspondiente (`/api/users/deleteuser/{idUser}` para usuarios normales o `/api/users/deleteprofuser/{idUser}` para usuarios profesionales).
2. El servidor procesa la solicitud, encuentra y elimina al usuario de la base de datos.
3. El servidor responde con un código HTTP 204 No Content, indicando que la operación se completó con éxito.

Integración en Aplicaciones

En una aplicación de administración, al seleccionar la opción para eliminar un usuario, la aplicación envía una solicitud DELETE al endpoint adecuado con el ID del usuario a eliminar. La aplicación maneja la respuesta para actualizar la interfaz de usuario y reflejar los cambios realizados en la base de datos.

Endpoints: Obtener Categorías

Descripción: Este endpoint permite a un cliente obtener las categorías de los servicios. Se pueden obtener todas las categorías o una categoría por su id.

Endpoint:

- Todas las categorías: <http://127.0.0.1:8080/api/categories>
- Categoría por id: <http://127.0.0.1:8080/api/categories/{id}>
- Método: GET

Encabezados:

- SSH-ApiKey (requerido): La API key de la aplicación para autenticar la solicitud.

Ejemplo de Solicitud:

http

GET http://127.0.0.1:8080/api/categories

Headers:

SSH-ApiKey: test_api_key_12345

GET http://127.0.0.1:8080/api/categories/test_category_id/

Headers:

SSH-ApiKey: test_api_key_12345

Ejemplo de Respuesta:

json category List

```
[  
  {"id":"Category ID",  
   "name":"Category Name"},  
  {"id":"Category ID",  
   "name":"Category Name"},  
  ...  
]
```

Descripción de la Respuesta:

- **id: id de la categoría.**
- **name: Nombre de la categoria.**

json

```
{  
  "id":"Category ID",  
  "name":"Category Name"  
}
```

Casos de Uso:

- **Obtener todas las categorías:**
 - **Escenario:** Un usuario crea un nuevo servicio y debe asignar una categoría.
 - **Proceso:**
 - El usuario envía una solicitud GET al endpoint /api/categoría, incluyendo los encabezados SSH-ApiKey.
 - El servidor responde con las categorías disponibles.
- **Obtener datos de una categoría:**
 - **Escenario:** Una aplicación móvil necesita mostrar la categoría de un servicio.
 - **Proceso:**
 - El usuario envía una solicitud GET al endpoint /api/categoría/{id}/, incluyendo los encabezados SSH-ApiKey.
 - El servidor responde con la categoría que corresponde al id.

Endpoints: CRUD para gestionar servicios

Descripción: Este endpoint permite a un usuario profesional gestionar los datos de un servicio. Permite crear, leer, actualizar y Borrar un servicio.

Endpoint:

- Todos los servicios : <http://127.0.0.1:8080/api/services>
- Un servicio : <http://127.0.0.1:8080/api/services/{id}/>
- Método: GET, PUT, POST, DELETE

Encabezados:

- **SSH-ApiKey (requerido):** La API key de la aplicación para autenticar la solicitud
- **Ejemplo de Solicitud:**
- **Authorization (requerido):** El valor debe ser Basic seguido de las credenciales del cliente (email:password) codificadas en Base64.

Ejemplo de Valor para el Encabezado Authorization:

- Credenciales: john.doe@example.com:securepassword123
- Codificado en Base64: am9obi5kb2VAZXhhbXBsZS5jb206c2VjdXJlcGFzc3dvcmQxMjM=

http

GET, POST <http://127.0.0.1:8080/api/services>

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Basic am9obi5kb2VAZXhhbXBsZS5jb206c2VjdXJlcGFzc3dvcmQxMjM=

GET, PUT, DELETE

<http://127.0.0.1:8080/api/services/{id}/>

Headers:

SSH-ApiKey: test_api_key_12345

Authorization:

Basic am9obi5kb2VAZXhhbXBsZS5jb206c2VjdXJlcGFzc3dvcmQxMjM=

Body:

```
{  
  "name": "John",  
  "note": "5",  
  "distance": "10",  
  "categoryID": "355e16b4-dfce-4083-89a1-95223fd22814",  
  "profUserID": "fde81cec-8e33-4766-bd60-871d65041b6c"  
}
```

Ejemplo de Respuesta:

json

```
{  
  "name": "service name",  
  "note": "note number",  
  "distance": "distance number",  
  "categoryID": "category id number",  
  "profUserID": "prof user id number"  
}
```

Descripción de la Respuesta:

- **name:** Nombre del servicio.
- **note:** Calificación del servicio.
- **distance:** Distancia al servicio.
- **categoryID:** Id de la categoría del servicio.
- **profUserID:** Id del usuario profesional.

Casos de Uso:

- **Crea un servicio:**
 - Escenario: Un usuario crea un nuevo servicio.
 - Proceso:
 - El usuario profesional envía una solicitud POST al endpoint /api/services, incluyendo los encabezados SSH-ApiKey y Authorization con las credenciales codificadas en Base64.
 - El servidor responde con una copia del servicio creado.
- **Editar un servicio:**
 - Escenario: Un usuario edita uno de los servicios.
 - Proceso:
 - El usuario profesional envía una solicitud PUT al endpoint /api/services/{id}/, incluyendo los encabezados SSH-ApiKey y Authorization con las credenciales codificadas en Base64.
 - El servidor responde con una copia del servicio editado.
- **Borra un servicio:**
 - Escenario: Un usuario elimina un servicio.
 - Proceso:
 - El usuario profesional envía una solicitud DELETE al endpoint /api/services/{id}/, incluyendo los encabezados SSH-ApiKey y Authorization con las credenciales codificadas en Base64.
 - El servidor responde con una copia del servicio borrado.

Ejemplo de Respuesta:

json

```
[  
  { "name": "service name",  
    "note": "note number",  
    "distance": "distance number",  
    "categoryID": "category id number",  
    "profUserID": "prof user id number" }, ...  
]
```

Casos de Uso:

- **Obtener un servicio:**
 - Escenario: Una aplicación móvil muestra todos los servicios.
 - Proceso:
 - El usuario profesional envía una solicitud **DELETE** al endpoint `/api/services`, incluyendo los encabezados `SSH-ApiKey` y `Authorization` con las credenciales codificadas en Base64.
 - El servidor responde con una copia de todos los servicios.
- **Obtener un servicio:**
 - Escenario: Una aplicación móvil muestra un servicio.
 - Proceso:
 - El usuario profesional envía una solicitud **GET** al endpoint `/api/services/{id}/`, incluyendo los encabezados `SSH-ApiKey` y `Authorization` con las credenciales codificadas en Base64.
 - El servidor responde con una copia del servicio.

Descripción de la Respuesta:

- **name:** Nombre del servicio.
- **note:** Calificación del servicio.
- **distance:** Distancia al servicio.
- **categoryID:** Id de la categoría del servicio.
- **profUserID:** Id del usuario profesional.

8. Seguridad y Privacidad

Seguridad

1. Autenticación y Autorización:

- Utilización de tokens JWT (JSON Web Tokens) para autenticar y autorizar las solicitudes de los usuarios.
- Los endpoints requieren tokens válidos para acceder a los recursos, garantizando que solo usuarios autenticados puedan interactuar con la API.

2. Cifrado de Datos:

- Los datos sensibles, como contraseñas, se cifran antes de almacenarse en la base de datos.
- Utilización de HTTPS para cifrar la comunicación entre el cliente y el servidor, protegiendo los datos en tránsito contra interceptaciones.

3. Manejo de Claves de API:

- Las solicitudes deben incluir el encabezado SSH-ApiKey para validar la autenticidad de la aplicación que realiza la solicitud.
- Las claves de API deben ser almacenadas de manera segura y nunca deben ser expuestas en el código del cliente.

4. Protección Contra Ataques:

- Implementación de límites de tasa (rate limiting) para prevenir ataques de fuerza bruta.
- Validación de entradas y protección contra inyecciones SQL y XSS (Cross-site Scripting).

Privacidad

1. Protección de Datos Personales:

- Cumplimiento con las regulaciones de protección de datos, como el GDPR (Reglamento General de Protección de Datos) en la UE.
- Recolección mínima de datos personales necesarios para el funcionamiento del servicio.

2. Acceso Controlado a Datos:

- Los usuarios solo pueden acceder a sus propios datos, con mecanismos estrictos de control de acceso para proteger la información de otros usuarios.
- Los administradores tienen acceso a datos según sea necesario para la gestión y mantenimiento del sistema, bajo estrictos protocolos de seguridad.

3. Política de Retención de Datos:

- Implementación de políticas de retención de datos para asegurar que los datos personales no se mantengan más allá del tiempo necesario para el propósito original.

8. Mantenimiento y Soporte

- Actualizaciones y Parches:
 - Regularmente se realizan actualizaciones de software para mejorar la funcionalidad, seguridad y rendimiento del sistema.
 - Parches de seguridad se aplican de inmediato ante la detección de vulnerabilidades.
- Soporte Técnico:
 - Disponibilidad de soporte técnico a través de canales definidos (correo electrónico, teléfono, sistema de tickets).
 - Tiempo de respuesta garantizado según el nivel de urgencia del problema reportado.
- Monitorización y Alertas:
 - Monitorización continua del sistema para detectar y resolver problemas proactivamente.
 - Configuración de alertas para notificar al equipo de soporte sobre eventos críticos o inusuales.
- Documentación:
 - Documentación detallada y actualizada para desarrolladores, administradores y usuarios finales.
 - Guías de usuario, tutoriales y FAQs (Preguntas Frecuentes) disponibles para ayudar a resolver problemas comunes.

9. Roadmap y Futuras Mejoras

1. Mejoras en la Experiencia de Usuario:
 - Implementación de nuevas funcionalidades basadas en el feedback de los usuarios.
 - Optimización de la interfaz de usuario para una mejor accesibilidad y usabilidad.
2. Escalabilidad y Rendimiento:
 - Mejoras en la infraestructura para soportar un mayor número de usuarios y transacciones.
 - Optimización del rendimiento del sistema para tiempos de respuesta más rápidos y mayor eficiencia.
3. Integraciones Adicionales:
 - Desarrollo de integraciones con otras plataformas y servicios para expandir las capacidades del sistema.
 - API ampliadas para permitir una integración más fluida con aplicaciones de terceros.
4. Funciones Avanzadas de Seguridad:
 - Implementación de autenticación multifactor (MFA) para una mayor seguridad de las cuentas de usuario.
 - Mejoras en los algoritmos de cifrado y gestión de claves.

10. Preguntas Frecuentes (FAQ)

- **¿Cómo puedo registrarme en la plataforma?**
 - Puedes registrarte enviando una solicitud POST al endpoint /auth/signup con la información requerida. Consulta la documentación de los endpoints para más detalles.
- **¿Qué hago si olvido mi contraseña?**
 - Actualmente, puedes contactar al soporte técnico para asistencia con la recuperación de la contraseña. Próximamente, se implementará una funcionalidad de recuperación automática de contraseñas.
- **¿Cómo puedo actualizar mi información personal?**
 - Puedes actualizar tu información personal enviando una solicitud PUT al endpoint correspondiente con tu ID de usuario y los datos que deseas actualizar.
- **¿Es seguro usar la plataforma?**
 - Sí, implementamos medidas de seguridad robustas, incluyendo cifrado de datos, autenticación y autorización con tokens, y protección contra ataques comunes.
- **¿Cómo puedo eliminar mi cuenta?**
 - Puedes eliminar tu cuenta enviando una solicitud DELETE al endpoint /api/users/deleteuser/{idUser} o /api/users/deleteprofuser/{idUser} según corresponda.
- **¿Dónde puedo obtener soporte técnico?**
 - Puedes obtener soporte técnico contactando a nuestro equipo a través del correo electrónico, teléfono o sistema de tickets. Los detalles de contacto están disponibles en nuestra página web.
- **¿Qué sucede con mis datos personales?**
 - Tus datos personales se manejan conforme a nuestras políticas de privacidad y protección de datos, asegurando que se utilicen solo para los fines necesarios y se almacenen de manera segura.
- **¿Cómo puedo dar feedback sobre la plataforma?**
 - Puedes enviar tu feedback a través de nuestro formulario de contacto en la página web o directamente al equipo de soporte. Apreciamos tus comentarios y sugerencias para mejorar continuamente nuestro servicio.

¿Qué te ha aportado el
desarrollar este proyecto?

¿Qué has aprendido?

¿Qué es lo que no volverías
hacer de la misma manera?

¿Qué cosas seguirías
haciendo en el futuro para
mejorar el proyecto?