TUIKIGAI COLOMBIA - Manual Técnico y de Usuario

Versión: 1.0
Cliente: Capital Martech SAS
Desarrollador: NeuronaX SAS
Fecha de entrega: 16/05/2025

Índice

---

1. Introducción

Bienvenido a la documentación oficial de **TUIKIGAI Colombia**, una plataforma web interactiva que permite a los usuarios descubrir su propósito de vida a través del método japonés Ikigai. Este documento es una guía integral para usuarios, administradores y desarrolladores, cubriendo desde el uso básico hasta la evolución técnica del sistema.

---

2. Visión General y Propósito

TUIKIGAI es una SPA (Single Page Application) moderna, robusta y escalable, que permite a los usuarios:

- Explorar su Ikigai mediante preguntas clave y visualizaciones interactivas.
- Personalizar su experiencia y adquirir productos digitales relacionados.
- Realizar compras y regalos de experiencias de Ikigai, con integración de pagos segura.
- Disfrutar de una experiencia fluida, responsiva y visualmente atractiva.

Propósito:

- Brindar una herramienta digital confiable y atractiva para el autodescubrimiento y la reflexión personal.
- Facilitar la adquisición y regalo de experiencias personalizadas.

- Servir como base tecnológica sólida para futuras evoluciones, incluyendo inteligencia artificial y nuevas integraciones.

El dominio, hosting y distribución de la plataforma son provistos por **NeuronaX SAS**, garantizando alta disponibilidad y soporte profesional. La base de datos Firestore y la gestión de pagos están bajo control del equipo de TUIKIGAI, asegurando la privacidad y seguridad de los datos.

---

3. Arquitectura y Tecnologías

3.1. Diagrama de Arquitectura

```
Usuario
   │
   ▼
Frontend (Qwik + TailwindCSS + D3.js)
   │
   ▼
Servicios (APIs internas, Firestore, MercadoPago)
   │
   ▼
Backend (Google Apps Script - Mock para futuras integraciones)
   │
   ▼
Base de datos (Firestore)
```

3.2. Tecnologías Principales

Capa	Tecnología	Descripción
Frontend	Qwik, TailwindCSS, D3.js	SPA reactiva, estilos modernos, visualización interactiva
Backend	Google Apps Script (Mock)	Listo para futuras integraciones con Google Sheets y automatización
Base de datos	Firestore	Almacenamiento seguro y eficiente de datos de usuario y transacciones
Pagos	MercadoPago	Integración robusta para pagos y regalos
Seguridad	NeuronaX Turnstile	Protección anti-bots
Analytics	NeuronaX Web Analytics	Seguimiento de conversiones y comportamiento sin cookies

Nota: El stack está cuidadosamente seleccionado para maximizar la eficiencia, seguridad y escalabilidad, permitiendo futuras integraciones y mejoras sin comprometer la estabilidad actual.

---

4. Estructura del Proyecto

```
/
├── src/
│   ├── components/        # Componentes UI y secciones principales
│   │   ├── sections/      # Secciones principales (Ikigai, Footer, Flujos)
│   │   ├── ui/            # Componentes de UI reutilizables
│   │   └── router-head/   # Configuración del head HTML
│   ├── routes/            # Rutas y páginas de la aplicación
│   │   ├── index.tsx      # Página principal
│   │   ├── payment/       # Rutas de pagos (success, pending, failure)
│   │   ├── privacidad/    # Política de privacidad
│   │   └── terminos/      # Términos y condiciones
│   ├── services/          # Lógica de negocio e integraciones
│   ├── utils/             # Funciones auxiliares y helpers
│   ├── types/             # Definiciones de tipos TypeScript
│   ├── styles/            # Estilos globales (TailwindCSS)
│   └── firebase.ts        # Configuración de Firestore
├── public/                # Recursos estáticos (imágenes, documentos)
├── backend/               # Scripts para futuras integraciones (Google Apps
Script)
├── utils/                 # Mock de integración con Google Sheets
├── dist/                  # Build de producción
├── scripts/               # Scripts de utilidad
├── package.json           # Dependencias y scripts de npm
└── README.md              # Documentación principal
```

4.1. Descripción de Carpetas y Archivos Clave

- **src/components/sections/UnifiedIkigai.tsx**: Núcleo de la experiencia Ikigai, lógica de visualización y manipulación de respuestas.
- **src/components/sections/PurchaseFlow.tsx**: Flujo de compra y regalo, integración con MercadoPago.
- **src/services/MercadoPagoService.ts**: Lógica de integración con la pasarela de pagos.
- **src/services/FirestoreService.ts**: Abstracción para operaciones con Firestore.
- **src/services/GoogleSheetsService.ts**: Listo para futuras integraciones reales con Google Sheets.
- **src/services/SiigoPaymentService.ts**: Mock para facturación electrónica DIAN.
- **utils/googleSheetUtils.js**: Mock de integración con Google Sheets.
- **backend/tuikigai-backend.js**: Script de Google Apps Script para futuras integraciones.

Recomendación: Mantener la estructura modular y la documentación interna de cada módulo para facilitar la evolución y el mantenimiento.

---

5. Flujos de Usuario y Funcionalidades

5.1. Ikigai Playground

- El usuario responde las 4 preguntas clave del Ikigai.
- El diagrama interactivo se actualiza en tiempo real usando D3.js.
- Se pueden personalizar colores y guardar la experiencia.

5.2. Proceso de Compra y Regalo

- Opción de comprar la experiencia para sí mismo o regalarla.
- Integración con MercadoPago para pagos seguros.
- Generación y canje de códigos de regalo.
- Confirmación y seguimiento de transacciones.

5.3. Canje de Código de Regalo

- El destinatario ingresa el código recibido.
- Validación automática contra Firestore.
- Acceso a la experiencia personalizada.

5.4. Administración y Seguridad

- Protección anti-bots con NeuronaX Turnstile.
- Análisis de tráfico y conversiones con NeuronaX Web Analytics.
- Gestión segura de datos y cumplimiento de normativas.

Diagrama de Flujo Simplificado:

```
[Inicio] → [Responde Ikigai] → [Visualiza Diagrama] → [Compra/Regala] → [Pago] →
[Código de Regalo] → [Canjea] → [Fin]
```

---

6. Instalación y Configuración

6.1. Requisitos Previos

- Node.js 18.x o superior
- npm 8.x o superior
- Cuenta de Google y Firebase (para futuras integraciones)
- Cuenta de MercadoPago (para pagos)
- Cuenta de Siigo (opcional, para facturación electrónica en desarrollos futuros)

6.2. Instalación

```
git clone https://github.com/NeuronaX-SAS/tuikigaicolombiacom.git
cd tuikigaicolombiacom
npm install
```

6.3. Configuración de Variables de Entorno

Crea un archivo `.env` en la raíz del proyecto con:

```
VITE_BACKEND_URL=https://script.google.com/macros/s/tu-script-id/exec
VITE_TURNSTILE_SITEKEY=tu-sitekey-de-neuronax-turnstile
VITE_MP_PUBLIC_KEY=tu-clave-publica-de-mercadopago
```

Importante: Nunca expongas tus claves o credenciales en el repositorio. Usa siempre variables de entorno.

---

7. Desarrollo, Mantenimiento y Mejora Continua

7.1. Desarrollo Local

```
npm run dev
```

Accede a `http://localhost:3000`.

7.2. Simulación de Producción

```
npm run serve
```

Simula el entorno de producción localmente.

7.3. Construcción para Producción

```
npm run build
```

El resultado estará en el directorio `dist/`.

7.4. Despliegue

El despliegue y la distribución están gestionados por NeuronaX SAS. Para actualizaciones o despliegues, contactar al equipo de soporte.

7.5. Actualización de Dependencias

```
npm outdated
npm update
```

Para actualizaciones importantes:

```
npm install package@latest
```

7.6. Diagnóstico de Problemas Comunes

- **Problemas de integración con Google Sheets**: Verifica credenciales, estructura y logs en Apps Script.
- **Problemas con pagos**: Revisa credenciales de MercadoPago y URLs de redirección.
- **Problemas de rendimiento**: Optimiza imágenes, usa Lighthouse y revisa la consola.

7.7. Auditoría de Seguridad

- Mantén dependencias actualizadas.
- Verifica vulnerabilidades con `npm audit`.
- Asegura que las claves API y tokens estén seguros y no expuestos.

7.8. Mejora Continua

- Documenta cada cambio relevante en el repositorio.
- Realiza revisiones de código periódicas.
- Mantén una cultura de pruebas y validación antes de cada despliegue.
- Considera la retroalimentación de usuarios para priorizar nuevas funcionalidades.

---

8. Integraciones y Servicios

8.1. Firestore

- Almacena respuestas de Ikigai, datos de pago y códigos promocionales.
- Gestión y acceso controlado por el equipo de TUIKIGAI.

8.2. MercadoPago

- Procesamiento de pagos y generación de códigos de regalo.
- Configuración de credenciales y URLs de redirección en el archivo `.env`.

8.3. Google Apps Script y Google Sheets

- Actualmente en modo mock para pruebas y desarrollo.
- Listo para futuras integraciones reales.
- Para activar la integración real, reemplazar `utils/googleSheetUtils.js` por una implementación real y asegurar el almacenamiento seguro de credenciales.

8.4. Siigo

- Integración mock para facturación electrónica DIAN.
- Listo para desarrollos futuros según requerimientos de Capital Martech SAS.

8.5. Seguridad y Analytics

- NeuronaX Turnstile y Web Analytics integrados para protección y análisis avanzado.

Nota: Las integraciones están diseñadas para ser desacopladas y fácilmente reemplazables o ampliables según las necesidades futuras del negocio.

---

9. Seguridad, Buenas Prácticas y Auditoría

- Mantén las dependencias actualizadas (`npm outdated`, `npm update`).
- Realiza auditorías de seguridad periódicas (`npm audit`).
- Nunca expongas claves o credenciales en el repositorio.
- Usa variables de entorno para información sensible.
- Realiza pruebas exhaustivas antes de activar nuevas integraciones.
- Documenta cualquier cambio relevante en la arquitectura o dependencias.
- Contacta a NeuronaX SAS para reportes de analytics o soporte avanzado.

Recomendación: Implementa revisiones de seguridad y pruebas automatizadas en cada ciclo de desarrollo.

---

10. Preparación para Integración de Inteligencia Artificial

La arquitectura de TUIKIGAI está preparada para evolucionar hacia una experiencia de Ikigai dinámico potenciada por inteligencia artificial (IA). A continuación, se detallan los módulos y puntos de extensión recomendados para futuras integraciones de IA:

10.1. Puntos de Integración Sugeridos

- **src/components/sections/UnifiedIkigai.tsx**
  - Aquí se gestiona la lógica de visualización y procesamiento de las respuestas del usuario. Para IA, se puede:
    - Incorporar un servicio que analice las respuestas y sugiera insights personalizados.
    - Integrar un modelo de lenguaje (API externa o local) para generar recomendaciones o reflexiones automáticas.
    - Añadir un botón o trigger para "Ikigai Dinámico" que consulte un endpoint de IA.
- **src/services/api.ts**
  - Este archivo puede extenderse para incluir llamadas a servicios de IA (por ejemplo, OpenAI, Azure AI, Google AI, etc.).
  - Ejemplo de función a agregar:

```
export async function getIkigaiAIInsights(respuestas) {
  // Llamada a API de IA
  // return await axios.post('https://api.tu-ia.com/ikigai',
respuestas);
}
```

- **src/routes/api/**
  - Puedes crear un endpoint local (por ejemplo, `src/routes/api/ikigai-ai.ts`) que reciba las respuestas del usuario, las procese y devuelva recomendaciones generadas por IA.
- **src/components/ui/**

- Aquí puedes crear componentes reutilizables para mostrar resultados, insights o visualizaciones generadas por IA.

10.2. Recomendaciones para la Integración de IA

- Mantener la separación de responsabilidades: la lógica de IA debe estar desacoplada de la UI.
- Usar servicios y hooks personalizados para consumir la IA.
- Documentar exhaustivamente cualquier integración de IA, incluyendo dependencias, endpoints y ejemplos de uso.
- Realizar pruebas exhaustivas y validaciones de seguridad, especialmente si se usan APIs externas.
- Considerar la privacidad y el consentimiento del usuario para el procesamiento de datos.

10.3. Ejemplo de Flujo para Ikigai Dinámico con IA

1. El usuario responde las preguntas del Ikigai.
2. Se envían las respuestas a un endpoint de IA (local o externo).
3. La IA procesa y devuelve insights personalizados.
4. Se muestran los resultados en un nuevo componente de la UI.

Ventaja Competitiva: Esta preparación permite que el proyecto evolucione hacia experiencias personalizadas y escalables, alineadas con tendencias globales de digitalización y bienestar.

---

11. Preguntas Frecuentes

¿Cómo puedo modificar la experiencia de usuario o el diseño?
Edita los componentes en `src/components/sections` y los estilos en `src/styles/main.css` o usando TailwindCSS.

¿Cómo agregar nuevas funcionalidades?
Crea nuevos servicios en `src/services` y componentes en `src/components`. Sigue la estructura modular y usa TypeScript para mantener la robustez.

¿Cómo activar la integración real con Google Sheets o Siigo?
Consulta la sección 8.3 y 8.4. Contacta a NeuronaX SAS para soporte y acompañamiento en la integración.

¿Cómo escalar o migrar la base de datos?
Firestore es escalable y gestionado por TUIKIGAI. Para migraciones, contacta al equipo técnico.

¿Cómo integrar inteligencia artificial para el Ikigai dinámico?
Consulta la sección 10 para recomendaciones y puntos de extensión.

¿Qué debo hacer si encuentro un error inesperado?
Revisa la consola del navegador, los logs de la base de datos y los endpoints de la API. Si el problema persiste, contacta al soporte técnico.

¿Cómo puedo contribuir a la mejora del proyecto?
Sugiere mejoras, reporta bugs o solicita nuevas funcionalidades a través del canal de soporte.

---

12. Soporte y Contacto

Para soporte técnico, mantenimiento o evolución del proyecto:

- **Email:** neuronax.sas@gmail.com & jaop.neuronax@gmail.com
- **Teléfono:** +57 320 3478322 (NeuronaX SAS)
- **Repositorio:** https://github.com/NeuronaX-SAS/tuikigaicolombiacom

---

13. Cierre y Recomendaciones

TUIKIGAI Colombia es una plataforma robusta, segura y lista para evolucionar según las necesidades de Capital Martech SAS. Se recomienda mantener la relación con NeuronaX SAS para futuras mejoras, integraciones y soporte, asegurando la continuidad y calidad del proyecto.

Recomendaciones Finales:

- Mantén la documentación actualizada.
- Prioriza la seguridad y la privacidad de los datos.
- Evalúa periódicamente nuevas tendencias tecnológicas para mantener la plataforma competitiva.
- Considera la integración de IA y nuevas experiencias digitales como parte de la evolución natural del producto.

---

14. Autoría y Firma

Departamento de Tecnología
NeuronaX SAS
Desarrollado para Capital Martech SAS
Versión 1.0 - 16/05/2025

¿Deseas personalizar o evolucionar la plataforma?
¡Cuenta con el respaldo y experiencia de NeuronaX SAS, desde 1983 le aportamos a Colombia!