MANUAL TÉCNICO

Sistema: SIAG – TechNova Solutions

Versión del manual: 1.0

Fecha: 28/08/2025

Integrantes:

Iván Madera

Brian Ortega

Camilo Cassiani

Sebastián Cera

Said Coley

Índice:

■ Propósito

■ Alcance

■ Componentes del sistema

3.1 Frontend Web (Cliente)

3.3 Backend (API)

3.4 Base de Datos

3.5 Integraciones externas

■ Requisitos técnicos

4.1 Backend

4.2 Variables de entorno

■ Diccionario de datos (Rails)

6.1 Ejemplo de formato

6.2 Cómo obtener la información

■ Relaciones principales

7.1 Convenciones de Rails

7.2 Comandos para ver relaciones

■ Código fuente

8.1 Estructura del repositorio

8.2 Qué incluir en el manual

8.3 Utilidades de inspección

■ Despliegue

9.1 Entornos

9.2 Pipeline de despliegue

■ Procedimientos de escalado (capacidad)

10.1 Backend Rails

10.2 Base de datos

10.3 Frontend

■ Apéndices

15.1 Comandos útiles

15.2 Ejemplo de endpoint

15.3 Glosario

1. Propósito:

Este documento describe la arquitectura técnica, componentes, relaciones de datos, lineamientos de código y procedimientos operativos del sistema **SIAG**. Sirve como guía para desarrolladores, DevOps y personal de soporte.

2. Alcance:

Objetivo general:

Documentar la arquitectura, componentes, configuración y operación técnica del sistema SIAG – TechNova Solutions para habilitar su mantenimiento, evolución y soporte por parte de desarrollo, DevOps y mesa de ayuda.

Objetivos específicos:

- Describir la arquitectura y los componentes del sistema (cliente, API/servidor, base de datos e integraciones).
- Documentar los procesos de negocio soportados y su flujo a nivel técnico.
- Presentar el modelo de datos y el diccionario de las entidades clave.
- Establecer requisitos técnicos, variables de entorno y consideraciones de seguridad.
- Definir lineamientos de despliegue, operación y escalado.
- Proveer referencias de código fuente y estructura del repositorio para facilitar el soporte de primer y segundo nivel.

Límites del documento (no cubre):

- Manual de usuario final (operación paso a paso por perfiles); se gestiona en documento aparte.
- Políticas corporativas detalladas de seguridad de la información.
- Credenciales reales o datos sensibles (se usan placeholders y gestión por entorno).
- Aspectos contractuales/legales ajenos al software.

Público objetivo:

Desarrolladores, DevOps, soporte N1/N2 y administradores de la plataforma.

- Componentes y arquitectura general (web, backend, base de datos, servicios auxiliares).
- Diccionario de datos (modelo relacional en Rails).
- Lineamientos de código fuente y estructura del repositorio.

3. Componentes del sistema:

3.1 Frontend Web (Cliente)

- **Framework:** Ruby on Rails.

- **Comunicación:** REST/JSON contra API backend.
- **Autenticación:** JWT

3.2 Backend (API)

- **Framework:** Ruby on Rails.
- **Versión:** Rails 8.0 sobre Ruby 3.4.5.
- Expone endpoints internos y vistas HTML integradas.

3.3 Base de datos

- sqlite.

3.3 Integraciones externas

- **Pasarela de pago:** (PayU).

4. Descripción de procesos:

Esta sección detalla los procesos de negocio que soporta SIAG a nivel técnico. Para cada proceso se indican actores, precondiciones, entradas, salidas, reglas principales y puntos de integración.

4.1 Autenticación y autorización

- Actores: Usuario final, Administrador.
- Resumen: Inicio de sesión con credenciales; emisión y validación de JWT para proteger endpoints.
- Precondiciones: Usuario registrado y activo.
- Entradas: email/usuario, contraseña.
- Salidas: Token JWT, perfil/roles cargados en sesión.
- Reglas: Política de contraseñas; expiración/renovación de tokens; rate limiting de intentos fallidos.
- Integración: Endpoints /auth/login, /auth/refresh (proteger todas las rutas privadas).

4.2 Gestión de catálogo (Categorías y Productos)

- Actores: Administrador (CRUD), Usuario (consulta).
- Resumen: Alta/baja/modificación de categorías y productos; carga de imágenes con ActiveStorage; manejo de precio y estado.
- Entradas: nombre, descripción, categoría, precio, imagen.
- Salidas: Producto publicado; image_url disponible en API.
- Reglas: Nombre único por categoría; validación de tamaños de imagen; precios enteros (centavos).
- Datos: products, categories (ver Diccionario de datos).

4.3 Carrito y checkout

- Actores: Usuario autenticado.
- Resumen: Construcción de carrito; cálculo de totales (sub-total, impuestos y/o envío).

- Precondiciones: Usuario con sesión activa.
- Reglas: Stock disponible al confirmar; precios tomados del catálogo vigente.

4.4 Pagos (Pasarela PayU)

- Actores: Usuario, Pasarela de pago.
- Resumen: Redirección a la pasarela, recepción de notificación/confirmación y actualización del estado de la orden.
- Estados: PENDING, APPROVED, DECLINED (reintento permitido).
- Seguridad: Nunca almacenar llaves reales en el repositorio; usar variables de entorno (ej.: PAYU_API_KEY, PAYU_MERCHANT_ID, PAYU_ACCOUNT_ID).
- Integración: URL de retorno y de confirmación (webhook) configuradas por entorno.

4.5 Órdenes y facturación básica

- Actores: Usuario, Administrador.
- Resumen: Generación de Orden desde el checkout; numeración; cambio de estados; emisión de comprobante y notificación por correo.
- Reglas: La orden queda en CREATED y transiciona según respuesta de pago.

4.6 Administración de usuarios y roles

- Actores: Administrador.
- Resumen: Alta/baja de usuarios, asignación de roles/permisos.
- Reglas: Sólo administradores pueden modificar roles.

4.7 Reportes operativos

- Actores: Administrador.
- Resumen: Ventas por rango de fechas, productos más vendidos, pedidos por estado.

- **Comunicación:** REST/JSON contra API backend.

4.1 Backend

- Ruby >= 3.4.5, Rails >= 8.0.0.
- Base de datos (SQLITE).

4.2 Variables de entorno

```
api_key = "4Vj8eK4rloUd272L48hsrarnUA"
```

5. Diccionario de datos:

Fuente: `db/schema.rb`

5.1 Ejemplo de formato

Tabla: `products`

📖 Descripción: Catálogo de productos.

📖 Campos:

- `id` (PK, integer/bigint, not null) → Identificador único del producto.
- `name` (string, not null) → Nombre del producto (ej. *Camisetas, Tazas*).
- `descripción` (text, nullable) → Descripción detallada del producto.
- `price` (integer, not null) → Precio del producto en centavos (ej. 10000 = \$100.00).
- `size` (string, nullable) → Talla o tamaño (en caso de aplicar).
- `image` (ActiveStorage attachment, nullable) → Imagen del producto. Se expone en la API como `image_url`.
- `created_at` (datetime, not null) → Fecha de creación del registro.
- `updated_at` (datetime, not null) → Última actualización del registro.

Índices:

- `index_products_on_name` (para búsquedas por nombre).

Fks:

- `category_id` → `categories.id` (relación con categorías de productos).

5.2 Como obtener la información

- Revisa `db/schema.rb` para tipos e índices.
- En SQLite: `bin/rails dbconsole` → `schema products`.

6. Relaciones principales:

6. Relaciones principales

- **1: N:** `has_many` / `belongs_to` (FK en la tabla "N").
- **N: N:** `has_and_belongs_to_many`.

- **1:1:** has_one / belongs_to.

7. Relaciones principales:

7.1 Estructura del repositorio

```
app/  
  controllers/  
  models/  
  views/  
  services/  
  policies/  
  uploaders/  
  
config/  
  environments/  
  credentials/  
  
db/  
  migrate/  
  schema.rb
```