CHIBCHAWEB HOSTING PLATFORM
APLICACIÓN PARA UNA COMPAÑÍA DE HOSPEDAJE DE PÁGINAS WEB

MANUAL TÉCNICO

Grupo A

Ingenieros:
Andrés José Acevedo Ardila
Brayan Steven Sánchez Casilima
Christian Camilo Lancheros Sánchez
Juan Esteban Ordoñez Sandoval
Karen Tatiana Bravo Rodríguez
Luis Felipe Mayorga Tibaquicha

Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería
Ingeniería de sistemas

Bogotá D.C., Agosto 2025

ChibchaWeb
Hosting Platform

Grupo A

Manual Técnico ChibchaWeb

2025

Índice

Índice de figuras

Índice de cuadros

1 Introducción

Este manual proporciona la información necesaria para la instalación, configuración, operación y mantenimiento del sistema ChibchaWeb, una plataforma web orientada al servicio de hospedaje de sitios web. El sistema permite la gestión de cuentas de clientes, distribuidores y empleados, la adquisición de dominios, facturación automática, administración de tickets de soporte, y más. Esta guía está dirigida a desarrolladores, administradores del sistema y personal técnico encargado del mantenimiento de la solución.

El manual técnico documenta el desarrollo e instalación de un sistema web que automatiza las operaciones clave de ChibchaWeb, incluyendo:

1. Registro y gestión de cuentas de clientes, empleados y distribuidores.
2. Validación de información sensible como direcciones y tarjetas de crédito.
3. Procesamiento de pagos mediante tarjetas aceptadas (VISA, MASTERCARD, DINERS).
4. Administración de dominios y envío de solicitudes a registradores externos.
5. Registro y gestión de tickets de soporte técnico.
6. Cálculo de comisiones para distribuidores y generación de archivos de transferencia.

El objetivo de esta documentación es proporcionar una guía técnica completa que facilite la instalación, configuración, uso, mantenimiento y actualización del sistema para desarrolladores, administradores del sistema y personal técnico de soporte.

2 Alcance del manual

Este documento está dirigido a desarrolladores, administradores de sistemas y personal de soporte que requieran instalar, configurar, mantener y operar la plataforma de gestión de servicios de ChibchaWeb. El alcance del manual incluye instrucciones detalladas para:

1. La instalación del entorno de desarrollo y producción.
2. La configuración de los archivos clave del sistema.
3. La conexión con la base de datos y los servicios externos.
4. La descripción funcional de cada componente del sistema.
5. Las operaciones necesarias para ejecutar y mantener la plataforma.

El sistema está compuesto por una arquitectura web basada en tecnologías modernas que permiten escalabilidad, claridad en la gestión de roles y eficiencia en la administración de dominios, pagos y soporte técnico.

2.1 Frontend

- **Framework**: Flask (utilizado también como microservidor web para el desarrollo). [2]
- **Estilos y UI:** Bootstrap. [3]

2.2 Backend

- **Lenguaje**: Python.
- **Framework principal**: Flask

2.3 Base de Datos

- **MySQL** gestionada y visualizada a través de DBeaver como cliente de administración. [1]

2.4 Control de Versiones y Colaboración

- **GitHub:** para manejo de versiones, control de ramas y colaboración entre el equipo.

2.5 Servicios Externos

- **OpenAI API:** para la automatización en la respuesta a tickets de soporte técnico.

3 Requisitos

3.1 Requisitos del Hardware

- **RAM:** 4GB mínimo, 16 GB ideal.
- **CPU**: 2 o más núcleos.
- **Almacenamiento**: 4 GB mínimo, 10 GB ideal.

3.2 Requisitos de Software

- **Sistema operativos:**
  - Windows 10/11.
  - Linux
- **Navegador**: Para que el aplicativo web en desarrollo funcione correctamente el navegador que el usuario use debe tener como requisito las siguientes características:
  - Soporte JavaScript ES6+: React, JSX.
  - Soporte Web APIs modernas.
  - Soporte CSS: Grid/Flexbox.
  - Soporte HTML5.

  Como recomendación se dará un listado de la versión mínima recomendada para los navegadores más usados:
  - Google Chrome: 100+
  - Mozilla: 100+
  - Microsoft Edge: 100+
  - Safari (macOS): 15+
  - Opera: 85+
  - Safari (iOS): 15+
  - Chrome/Edge Android: Última versión
- **Entorno de desarrollo**
  - Node.js: v18.
  - npm: v9.
  - Python: v3.12.
  - PIP: con las siguientes librerías para en Python.

    - PIP
    - fastapi
    - Uvicorn
    - requests
    - beautifulsoup4
    - sqlalchemy
    - pymysql
    - python-dotenv
    - email-validator
    - bcrypt
    - passlib
    - cryptography
    - reportlab
    - Openai
- IDE:
    - VS Code
    - Zed
- Base de datos
    - MySQL
- Gestor base de datos:
    - DBeaver
- Bundle
    - Vite
- Control de versiones:
    - Git

```
CREATE DATABASE chibchaweb;
```

Figura 1: Crear base de datos.

```
CREATE USER 'chibcha'@'localhost' identified by 'chibcha';
```

Figura 2: Crear usuario en base de datos.

4 Instalación

Para realizar la instalación se recomenda crear una carpeta general para el proyecto, dentro de esta carpetas separadas por cada lógica.

4.1 Base de datos

Para realizar la instalación de la base de datos se cuenta con las siguientes dos opciones:

- Instalación local la cual permite realizar cambios que no afectan la lógica de desarrollo ni la de producción.
- Instalación producción la cual permite actualizar la base de datos que se comunica con todos los clientes del aplicativo.

Esta instalación es la que debe usarse para las diferentes etapas del desarollo de cualquier nueva funcionalidad.

Para realizarse se deben seguir los siguientes procesos:

Instalación MySQL

1. Instalar MySQL Server.
2. Crear la base de datos.
3. Crear un nuevo usuario.
4. Otorgar y actualizar privilegios al usuario.

```
GRANT ALL PRIVILEGES ON chibcha.* TO 'chibcha'@'localhost';
FLUSH PRIVILEGES;
```

Figura 3: Dar privilegios al usuario.

```
git clone https://github.com/Rocnarx/Chibchaweb-Hosting-Platform-
    Database.git
```

Figura 4: Clonar repo Database.

Configuración DBeaver

Ya instalado el Dbeaver se procede a crear nueva conexión en DBeaver:

1. Abrir DBeaver.
2. Clic en "New Database Connection" (ícono de plug).
3. Eligir MySQL.
4. Llena los campos:
   - Host: localhost
   - Port: 3306
   - Database: chibchaweb
   - Username: chibcha
   - Password: chibcha
5. Clic en Test Connection.
   - Permitir instalar el driver JDBC, solo aceptar y dejar que DBeaver lo descargue.
6. Clic en Finish.

Configuración Base de datos

1. Clonar el repositorio:
2. Ejecutar en Dbeaver el archivo CreacionDB.sql
3. Ejecutar en DBeaver el archivo DatosBasicos.sql

4.2 Backend

Configurar requirimientos

1. Descargar e instalar Python 3.12
   - Windows: Descargar de https://www.python.org/downloads/release/python-3120/
   - Linux:

```
sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository ppa:deadsnakes/ppa
sudo apt update
sudo apt install python3.12 python3.12-venv python3.12-dev
```

Figura 5: Instalar Python 3.12

```
    python3 -m venv ChibchaBack
```

Figura 6: Crear venv.

Configurar Back

1. En la carpeta back crear un entorno virtual
2. Iniciar el entorno virtual
3. Clonar el repositorio:
4. Instalar los paquetes requeridos que se encuentran en archivo requeriments.text:
5. Crear archivo .env el cual nos permetira conectar el back con los demás componentes
6. Configurar archivo .env con el siguiente texto

4.3 Front

Configurar requerimientos

1. Instalar Node.js y npm
    - Windows: Descargar e instalar desde nodejs.org (recomiendo la versión LTS)
    - Linux:

```
    sudo apt update
    sudo apt install nodejs npm
```

Figura 11: Instalar node.

```
    # -- Windows
    ./ChibchaBack/Scripts/activate
    #Linux
    source ChibchaBack/bin/activate
```

Figura 7: Activar venv

```
git clone https://github.com/Rocnarx/Chibchaweb-Hosting-
   Platform-Backend.git
```

Figura 8: Clonar repo backend.

```
pip install -r Chibchaweb-Hosting-Platform-Backend/
   requirements.txt
```

Figura 9: Instalar requerimientos back.

Configurar Front

1. Clonar el repositorio en la carpeta destinada a Front
2. Acceder a la carpeta de proyecto e instalar las dependencias

```
cd Chibchaweb-Hosting-Platform-Frontend
npm install
```

Figura 13: Instalar dependencias front.

3. Ejecutar el proyecto en desarrollo

```
npm run dev
```

Figura 14: Crear base de datos.

4. Crear archivo .env el cual nos permetira conectar el front con los demás componentes
5. Configurar archivo .env con el siguiente texto

```
VITE_API_URL=https://fastapi-app-production-d0f4.up.railway.app
VITE_API_KEY=dev-4b872f28-23f9-4a1e-9a60-1cb7f4090abc
```

Figura 15: Configurar .env front

```
FERNET_KEY= Solicitar al equipo de desarrollo
API_KEY= Solicitar al equipo de desarrollo
DATABASE_URL= mysql+pymysql://root:.. Solicitar al equipo de
   desarrollo
OPENAI_API_KEY= Solicitar al equipo de desarrollo
```

Figura 10: Editar .env back

```
git clone https://github.com/Rocnarx/Chibchaweb-Hosting-Platform-
   Frontend.git
```

Figura 12: Clonar repo Front.

5 Descripción de los módulos

5.1 Backend

El backend del proyecto fue desarrollado en Python 3.12 bajo FastAPI, utilizando SQLAlchemy ORM para la gestión de la base de datos MySQL, este fue dividido en varios módulos funcionales cómo son:

5.1.1 Módulo de Gestión de Carrito

Este módulo permite la gestión de dominios en carritos de compra por parte de la cuenta del cliente, sus endpoint principales se pueden ver en el siguiente cuadro:

Método	Endpoint	Descripción
POST	/agregarCarrito	Crear un nuevo carrito.
POST	/agregarDominioACarrito	Agregar un dominio al carrito.
PUT	/updateCarrito	Actualizar información del carrito.
GET	/carrito/obtener-por-cuenta	Obtener todos los carritos asociados a una cuenta.
DELETE	/eliminarDominioCarrito	Eliminar un dominio del carrito.
GET	/carrito/dominios	Obtener los dominios facturados.

Cuadro 1: Módulo de Gestión de Carrito

5.1.2 Módulo de Gestión de Dominios

Este módulo, cumple la funcionalidad de consultar y registrar dominios, así como verificar disponibilidad de estos y transferirlos por correo electrónico, a continuación se describen los principales endpoin de este:

Método	Endpoint	Descripción
POST	/DominiosDisponible	Verificar si un dominio está disponible.
POST	/agregarDominio	Agregar un nuevo dominio a la plataforma.
PUT	/ActualizarOcupadoDominio	Actualizar el estado ocupado/libre de un dominio.
POST	/dominios/agregar-a-carrito-existente	Agregar un dominio a un carrito ya creado.
PUT	/TransferenciaDominio	Transferir un dominio entre cuentas.
GET	/DominiosAdquiridos	Consultar dominios adquiridos por el usuario.
GET	/dominios/vigencia	Obtener la vigencia de los dominios adquiridos.

Cuadro 2: Módulo de Gestión de Dominios

5.1.3 Módulo de Cuentas y Autenticación

Este se centra en la funcionalidad de gestión de todas las cuentas del sistema (Cliente, distribuidor, administrador y empleado), donde se incluye, su registro, su ingreso y su autenticación segura.

Método	Endpoint	Descripción
POST	/login	Autenticación de usuario.
POST	/registrar2	Registro de nueva cuenta.
GET	/cuentas-por-tipo	Obtener cuentas filtradas por tipo.
POST	/cuenta-por-correo	Obtener datos de cuenta mediante correo electrónico.
POST	/solicitar-registro	Solicitar registro de cuenta.
GET	/confirmar-registro	Confirmar el registro de una cuenta.
GET	/estoy-verificado	Verificar si una cuenta está validada.
PUT	/admin/modificar-cuenta/idcuenta	Modificar datos de una cuenta (solo administrador).

Cuadro 3: Módulo de Cuentas y Autenticación

5.1.4 Módulo de Planes y Hosting

Método	Endpoint	Descripción
PUT	/CambiarPlan	Cambiar el plan de una cuenta.
GET	/Planes	Consultar planes disponibles.

Cuadro 4: Módulo de Planes y Hosting

5.1.5 Módulo de Soporte y Ticket

Método	Endpoint	Descripción
POST	/CrearTicket	Crear un nuevo ticket de soporte.
POST	/ticket/codigo/respuesta	Agregar una respuesta a un ticket existente.
GET	/consultarTicketporIDCUENTA	Consultar tickets abiertos por una cuenta.
PATCH	/CambiarNivelTicket/codigo	Cambiar el nivel de prioridad del ticket.
PATCH	/CambiarEstadoTicket/codigo	Cambiar el estado de un ticket.
PATCH	/asignarTicket/codigo	Asignar un ticket a un empleado.
GET	/ticket/codigo	Obtener detalles de un ticket específico.
GET	/ver-tickets	Consultar tickets por estado.
GET	/ver-tickets-niveles	Consultar tickets por estado y nivel de prioridad.

Cuadro 5: Módulo de Soporte y Ticket

5.2 Middleware de Automatización (OpenAI)

El sistema integra un middleware con OpenAI para:

- Clasificación automática de solicitudes entrantes (cancelación, reclamo, consulta, agradecimiento, solicitud, otra).
- Generación automática de respuesta preliminar y cordial al cliente (vía correo), notificando recepción del ticket y plazo estimado de respuesta (24 a 48 horas).
- Gestión de historial de tickets: todos los mensajes se almacenan como archivos JSON para trazabilidad.

Este componente utiliza el API de openrouter.ai y las claves definidas en las variables de entorno del backend (OPENAI_API_KEY).

5.3 Base de datos

La base de datos utilizada en el sistema es de tipo relacional, implementada sobre MySQL, con su gestión y visualización realizada a través de DBeaver. Su diseño responde a la lógica del negocio de un sistema de adquisición de dominios y hosting, integrando funcionalidades de carrito de compras, facturación, métodos de pago y gestión de usuarios.

A continuación, se describe brevemente la estructura general de las entidades más relevantes:

CUENTA: Representa a los usuarios registrados en el sistema. Almacena la información asociada a la cuenta del cliente.

CARRITO: Contiene los productos (dominios o paquetes) seleccionados por el usuario para la compra. Se relaciona con las tablas ESTADOCARRITO, CARRITODOMINIO y FACTURA.

DOMINIO: Define los nombres de dominio disponibles, incluyendo su precio y estado de ocupación.

CARRITODOMINIO: Tabla intermedia que relaciona los dominios añadidos al carrito.

FACTURA: Registra los pagos asociados a carritos finalizados, incluyendo el valor total, fecha y método de pago.

PAQUETEHOSING y INFOPAQUETEHOSING: Juntas describen los servicios de hosting disponibles, sus características técnicas (espacio en disco, certificados SSL, correos, etc.) y periodicidad de cobro.

FACTURAPAQUETE e ITEMPAQUETE: Vinculan los paquetes de hosting con las facturas emitidas, permitiendo registrar múltiples ítems por factura.

METODOPAGOCUENTA: Permite al usuario registrar métodos de pago, ya sea tarjeta u otro tipo, asociados a su cuenta.

TARJETA y TIPOTARJETA: Gestionan los datos de tarjetas de crédito o débito que pueden ser utilizadas como medio de pago.

TIPOMETODOPAGO: Catálogo que clasifica los diferentes tipos de métodos de pago.

TICKET: Módulo de soporte para generación y atención de incidencias o solicitudes por parte del cliente.

PAIS: Entidad referencial para identificar el país del usuario o cuenta.

6 Procedimientos de uso

A continuación, se detallan los procedimientos básicos para utilizar las funcionalidades del sistema ChibchaWeb, desde la perspectiva técnica. Este apartado no sustituye al manual de usuario final, sino que proporciona una guía operativa para verificar el correcto funcionamiento del sistema y realizar pruebas.

6.1 Acceso al sistema

1. Abrir el navegador y acceder al dominio principal: https://www.chibchaweb.site/
2. Iniciar sesión utilizando una cuenta registrada:
   (a) Para pruebas técnicas, se pueden usar cuentas de tipo administrador, distribuidor o empleado.
   (b) Los accesos están restringidos según el idtipocuenta.

6.2 Registro de nuevos usuarios

1. Desde el frontend, dirigirse al formulario correspondiente:
   (a) Registro de clientes
   (b) Registro de distribuidores
   (c) Registro de empleados
2. Llenar los campos requeridos:
   (a) Nombre, identificación, correo, teléfono, dirección, contraseña, tipo de cuenta, país y plan.
3. Enviar el formulario. El sistema realiza una solicitud POST a la API (/registrar2).
4. Verificar en la base de datos MySQL que los datos se almacenen correctamente en la tabla CUENTA.

6.3 Gestión de dominio y facturación

1. Iniciar sesión como usuario autorizado (cliente o distribuidor).
2. Navegar a la sección de dominios.
3. Realizar una compra o agregar un dominio al carrito.
4. Proceder con la facturación.
5. Verificar en la base de datos que se actualicen las tablas relacionadas:
   (a) DOMINIO, CARRITO, FACTURA, TRANSACCION.

6.4

Creación y seguimiento de tickets

1. Desde el frontend, acceder a la sección de soporte/tickets.
2. Crear un nuevo ticket describiendo el problema o requerimiento.
3. El sistema registra automáticamente la solicitud y, en segundo plano, utiliza la clave de OpenAI desde el backend para generar una respuesta automática inicial (si está habilitado).
4. Los tickets pueden ser gestionados desde el panel administrativo.

6.5 Verificación de autenticación y permisos

- La autenticación se maneja desde el backend mediante tokens.
- Los permisos se controlan con base en el idtipocuenta asignado.
- Verificar que:
    1. Usuarios con estado idestado = 5 no puedan iniciar sesión.
    2. Los formularios de login y registro validen correctamente los datos.

7 Glosario y anexos

A

- **API (Application Programming Interface):** Conjunto de definiciones y protocolos que permiten la comunicación entre distintas aplicaciones o servicios de software.
- **Administrador del sistema:** Usuario con permisos elevados para gestionar configuraciones, cuentas, dominios y operaciones críticas del sistema.
- **Automatización (OpenAI):** Integración de la API de OpenAI para generar respuestas automáticas a tickets de soporte mediante inteligencia artificial.

B

- **Bootstrap:** Framework de diseño front-end que facilita la creación de interfaces web responsivas y consistentes.
- **Bcrypt:** Algoritmo de cifrado utilizado para proteger contraseñas almacenadas de forma segura.
- **Bundler:** Herramienta que agrupa y optimiza archivos JavaScript, CSS y otros recursos para su uso en producción.

C

- **Carrito de compra:** Funcionalidad que permite a los usuarios seleccionar, almacenar y gestionar productos o servicios antes de completar una compra.
- **Cliente (cuenta):** Usuario registrado en el sistema que adquiere y gestiona servicios como dominios y hosting.
- **Comisión:** Porcentaje o monto fijo asignado a distribuidores por ventas realizadas dentro del sistema.
- **Cuenta (CUENTA):** Registro individual de usuario dentro del sistema, ya sea cliente, distribuidor, empleado o administrador.

D

- **DBeaver:** Herramienta de administración de bases de datos que facilita la visualización y edición de datos en entornos SQL.
- **Despliegue:** Proceso de publicación de la aplicación en un entorno accesible
- **Distribuidor:** Usuario con permisos especiales para revender servicios de dominio/hosting, gestionando clientes y comisiones.

- **Dominio (DOMINIO):** Nombre único que identifica un sitio en Internet, adquirido y gestionado por usuarios dentro del sistema.
- **DOMINIOSDisponible:** Tabla o entidad que registra los dominios libres para su compra o asignación.
- **Documentación técnica:** Conjunto de manuales, diagramas y guías destinados a facilitar el desarrollo, uso y mantenimiento del sistema.

E

- **Endpoint:** URL específica de una API que permite ejecutar operaciones mediante solicitudes HTTP (GET, POST, etc.).
- **Estado del carrito:** Condición actual del carrito de un usuario (abierto, confirmado, pagado, cancelado, etc.).
- **Estado del ticket:** Seguimiento del progreso de un ticket de soporte (pendiente, en proceso, resuelto, cerrado).
- **Empleado (cuenta):** Usuario con rol operativo dentro del sistema, encargado de soporte o administración técnica.

F

- FastAPI: Framework web para construir APIs en Python, utilizado en el backend por su velocidad y tipado moderno.
- Flask: Microframework web para Python utilizado para crear la interfaz cliente/administrador y manejar solicitudes HTTP.
- Frontend: Parte del sistema visible al usuario, desarrollada con tecnologías web como HTML, CSS, JS y React.

G

- **Git / GitHub:** Sistema de control de versiones y plataforma colaborativa para alojar el código fuente del sistema.
- **Gestión de tickets:** Funcionalidad que permite registrar, asignar y resolver solicitudes o incidencias de usuarios.
- **Gestión de dominios:** Módulo que permite registrar, transferir o administrar dominios vinculados a los clientes.
- **Gestión de cuentas:** Funcionalidad administrativa para crear, editar, suspender o eliminar cuentas del sistema.

H

- **Hosting:** Servicio de alojamiento web que permite publicar sitios y aplicaciones en Internet, con diferentes planes disponibles.

I

- **Identificación de cuenta:** Proceso que permite determinar el tipo, estado y permisos de una cuenta específica.
- **ItemPaquete:** Unidad o ítem que forma parte de un paquete de hosting, dominio u otro servicio agregado al carrito.

J

- **JSX (JavaScript XML):** Sintaxis utilizada en React para escribir componentes con estructura similar a HTML.

K

L

M

- **Middleware:** Función o componente intermedio que procesa solicitudes entre el cliente y el backend.
- **Método de pago:** Forma en que un cliente puede abonar por servicios (tarjeta, transferencia, etc.).
- **MySQL:** Sistema de gestión de bases de datos relacional utilizado por la aplicación para almacenar datos estructurados.

N

- **NPM (Node Package Manager):** Gestor de paquetes para Node.js, utilizado para instalar dependencias en el frontend.

O

- **OpenAI API:** Interfaz de programación utilizada para acceder a modelos de inteligencia artificial generativa de OpenAI.
- **OpenRouter.ai:** Plataforma que permite enrutar solicitudes a múltiples modelos de lenguaje, incluyendo los de OpenAI.

P

- **Paquete de hosting:** Conjunto de servicios agrupados bajo un plan que incluye almacenamiento, ancho de banda, correos, etc.
- **Passlib:** Biblioteca de Python utilizada para gestionar cifrado de contraseñas y autenticación segura.
- **Plan (de hosting):** Nivel de servicio contratado, con características técnicas y precio definido.
- **Python:** Lenguaje de programación principal utilizado en el backend del sistema.

Q

R

- **React:** Librería de JavaScript para construir interfaces de usuario dinámicas y componentes reutilizables.
- **Registro de cuenta:** Proceso por el cual un nuevo usuario se da de alta en el sistema.
- **ReportLab:** Biblioteca de Python utilizada para generar archivos PDF (por ejemplo, facturas o reportes).
- **Roles de usuario:** Clasificación de cuentas según sus privilegios (cliente, empleado, administrador, distribuidor, etc.).

S

- **Segmentación de cuentas:** Clasificación y asignación de cuentas según criterios administrativos o funcionales.
- **Soporte técnico:** Área dedicada a resolver consultas, problemas o solicitudes de los usuarios del sistema.
- **SQLAlchemy:** ORM de Python que facilita la interacción con bases de datos relacionales como MySQL.
- **Sistema de autenticación:** Mecanismo que valida la identidad de usuarios y otorga acceso según sus permisos.
- **Sistema operativo:** Software base que permite la ejecución del sistema, como Windows o Linux.

T

- **Tarjeta (de pago):** Medio electrónico utilizado por los clientes para abonar productos o servicios.
- **Ticket:** Solicitud registrada por un cliente relacionada con soporte técnico, facturación, etc.
- **Token:** Cadena generada para identificar sesiones, validar acciones o proteger datos en tránsito.
- **Transferencia de dominio:** Proceso mediante el cual un dominio cambia de registrador o de cuenta dentro del sistema.
- **Tipo de cuenta:** Clasificación de una cuenta según su función: cliente, empleado, distribuidor, etc.
- **Tipo de método de pago:** Categoría asignada al método usado para abonar: tarjeta, efectivo, transferencia.

U

- **Uvicorn:** Servidor ASGI utilizado para ejecutar aplicaciones web basadas en FastAPI.
- **Usuario final:** Persona que utiliza el sistema para contratar servicios y administrar su cuenta.

V

- **Vite:** Herramienta de desarrollo que acelera la carga y construcción de proyectos frontend modernos.
- **Visual Studio Code (VS Code):** Editor de código ampliamente utilizado por desarrolladores para programar y depurar.

W

- **Web APIs:** Conjunto de servicios accesibles mediante HTTP que permiten la integración entre componentes del sistema.
- **Windows 10/11:** Sistemas operativos compatibles con la instalación local y el desarrollo del sistema.

X

Y

Z

- **Zed:** Editor de código de Linux utilizado por desarrolladores para programar y depurar.

Referencias

[1] Oracle Corporation. Mysql documentation, 2025. https://dev.mysql.com/doc/, Accedido el 5 de agosto de 2025.

[2] Pallets Projects. Flask documentation, 2025. https://flask.palletsprojects.com/, Accedido el 5 de agosto de 2025.

[3] The Bootstrap Authors. Bootstrap documentation, 2025. https://getbootstrap.com/docs/, Accedido el 5 de agosto de 2025.