POS

SISTEMA POS RESTAURANTE

Documentación técnica e implementación

Tecnologías · Funcionalidades · Instalación · Despliegue web

CONTENIDO DEL DOCUMENTO

1. Tecnologías utilizadas
2. Funcionalidades del sistema
3. Instalación paso a paso en una computadora nueva
4. Hosting recomendado y comparativa
5. Despliegue del sistema en producción

GUÍA COMPLETA DE IMPLEMENTACIÓN

Para administradores y técnicos · Mayo 2026

1. TECNOLOGÍAS UTILIZADAS

El sistema está construido como una **aplicación web monolítica server-rendered** sobre el framework Laravel. La elección del stack prioriza estabilidad, una comunidad madura en español y compatibilidad con cualquier hosting PHP. No requiere servidor Node permanente: el frontend se compila a archivos estáticos y todo el HTML lo genera el servidor.

Arquitectura por capas



Detalle del stack

Capa	Tecnología	Versión	Propósito
Backend	**PHP**	8.3.16	Lenguaje de servidor
	Laravel	12.44	Framework MVC, routing, ORM, validación
	Composer	2.x	Gestor de dependencias PHP
Base de datos	**MySQL**	8.0+	Persistencia (usuarios, productos, ventas, SUNAT)
Frontend	**Blade**	—	Motor de plantillas Laravel
	Bootstrap	5.3	Framework CSS (UI responsive)
	Vite	7.0	Bundler y dev server
	Tailwind CSS	4.0	Utility-first CSS (complemento)
	Sass	1.97	Preprocesador CSS
	Bootstrap Icons	—	Iconografía

Facturación	**Greenter (suite)**	2.x	Emisión electrónica SUNAT Perú
	endroid/qr-code	5.1	Generación de códigos QR
	OpenSSL	—	Firma digital RSA-SHA1
Servidor web	**Apache / Nginx**	—	Servidor HTTP (incluido en Laragon/hosting)
	Laragon (dev)	6.0+	Stack PHP todo-en-uno para desarrollo

2. FUNCIONALIDADES DEL SISTEMA

El sistema es un **POS (Point of Sale) completo para restaurantes** con módulos para gestión de salones y mesas, ventas, cocina, inventario, caja, delivery, reservas, BI y facturación electrónica SUNAT. Tres roles operan en él: **administrador, cajero y mozo**, cada uno con acceso a las áreas que le corresponden.

Módulos operativos

Módulo	Funcionalidades
Punto de venta (POS)	• Mapa de mesas por salones con estados en vivo (libre, ocupada, lista) • Crear orden, añadir productos, modificar cantidades, agregar notas • Lector de código de barras • Aplicar descuento global, propina y dividir cuenta (split) • Mover mesa, pre-cuenta y ticket de cocina • Cobro con efectivo o tarjeta · cálculo de vuelto • Elegir tipo de comprobante: Ticket / Boleta / Factura
Cocina	• Monitor de cocina (KDS) con tickets pendientes • Cambio de estado por ítem: pendiente → preparando → listo → servido • Generación automática de ticket de cocina al confirmar pedido
Caja	• Apertura de caja con monto inicial • Cierre de caja con arqueo (efectivo, tarjeta, total ventas, gastos) • Historial de aperturas/cierres por turno • Bloqueo de POS si cajero no abrió caja
Productos e inventario	• CRUD de productos con foto, precio, costo, stock, código de barras • Categorías de productos • Recetas (escandallos): producto compuesto por ingredientes con descuento automático • Precio promocional, recomendado del chef, "nuevo" • Ajuste manual de stock con motivo • Kardex (historial de movimientos de inventario)
Mesas y salones	• Crear/editar/eliminar salones (Salón principal, Terraza, etc.) • Crear mesas con capacidad y coordenadas (mapa visual) • Drag & drop para reorganizar el mapa
Delivery	• Pedidos a domicilio con cliente, dirección, teléfono • Asignar repartidor, cambiar estado (recibido → en camino → entregado) • Cobro al entregar, anulación, listado de pedidos • CRUD de repartidores
Reservas	• Agenda de reservas con fecha, hora, número de personas y mesa • Cambio de estado: pendiente, confirmada, completada, cancelada
Clientes	• Base de datos de clientes con DNI/RUC, nombre, teléfono • Búsqueda rápida desde el POS
Gastos	• Registro de gastos del negocio (luz, agua, insumos, etc.) • Asociados a la caja del turno

Módulos administrativos y de análisis

Módulo	Funcionalidades
Dashboard	• Resumen del día: ventas, órdenes, ticket promedio • Gráficas de ventas por categoría • Estado de mesas en tiempo real
Reportes (BI)	• Ventas por rango de fechas • Ranking de productos más vendidos • Análisis por categoría y mesero • Exportación a Excel/PDF
Historial de ventas	• Listado de todas las órdenes con filtros • Reporte diario con totales por método de pago • Reimprimir ticket de venta
Facturación electrónica SUNAT	• Emisión automática de Boleta y Factura al cobrar • Listado de comprobantes con filtros y stats • PDF imprimible A4 + ticket 80 mm con QR • Notas de Crédito (catálogo 09) • Resumen Diario de Boletas • Reintento de envíos y descarga de XML/CDR • Configuración SUNAT en UI con upload .pfx
Usuarios y roles	• CRUD de usuarios con roles: admin, cajero, mozo • Permisos por rol enforcados por middleware
Configuración	• Datos de la empresa, dirección, teléfono • Logo, mensaje pie de ticket, zona horaria, moneda • Configuración SUNAT completa
Sistema	• Reset de datos (limpia ventas/órdenes manteniendo catálogo) • Backup manual de la base de datos • Restore desde backup
Menú digital público	• Página pública del menú (QR en mesa) • Categorías, productos con foto y precio • Marca "nuevo" y "recomendado del chef"

3. INSTALACIÓN EN COMPUTADORA NUEVA

Esta sección explica cómo dejar el sistema funcionando en una computadora nueva (Windows 10/11) desde cero. La forma recomendada usa **Laragon**, un paquete todo-en-uno gratuito que trae PHP, Apache, MySQL y herramientas en un solo instalador, evitando configuraciones manuales.

FLUJO DE INSTALACIÓN LOCAL (≈ 30 minutos)



Requisitos mínimos de la computadora

Componente	Mínimo recomendado
Sistema operativo	Windows 10 / 11 (64 bits)
Procesador	Intel Core i3 / AMD Ryzen 3 o superior
Memoria RAM	4 GB (8 GB recomendado)
Disco	5 GB libres (10 GB recomendado)
Conexión a internet	Necesaria para instalar y para enviar a SUNAT

Programas a descargar e instalar

1 Instalar Laragon (incluye PHP, Apache, MySQL)

Descarga la versión **Full** desde `laragon.org/download`. Ejecuta el instalador con permisos de administrador. Acepta los valores por defecto. Al terminar, Laragon ya trae PHP 8.3, Apache, MySQL 8 y phpMyAdmin listos para usar. Pulsa **Start All** en la ventana de Laragon para arrancar los servicios.

2 Habilitar extensiones de PHP necesarias

Click derecho en el icono de Laragon → **PHP** → **Extensions** y asegúrate de que estén marcadas: `soap`, `openssl`, `zip`, `xml`, `dom`, `mbstring`, `curl`, `intl`, `fileinfo`, `pdo_mysql`, `gd`. La extensión SOAP es crítica para SUNAT.

3 Instalar Composer (gestor de dependencias PHP)

Laragon ya trae Composer integrado. Para verificarlo, abre PowerShell en cualquier carpeta y ejecuta `composer --version`. Si no aparece, descárgalo de `getcomposer.org/download` e instálalo (selecciona el PHP de Laragon cuando lo pida: `C:\laragon\bin\php\php-8.3.x`).

4 Instalar Node.js (para compilar el frontend)

Descarga la versión **LTS** desde `nodejs.org` (recomendado: Node 20+). Ejecuta el instalador con valores por defecto. Verifica con `node --version` y `npm --version`.

5 Instalar Git (control de versiones)

Descarga desde `git-scm.com/download/win`. Acepta valores por defecto. Permite clonar el proyecto y mantenerlo actualizado.

6 Obtener el código fuente

Si tienes el proyecto en un repositorio Git, abre PowerShell en la carpeta `C:\laragon\www` y ejecuta: `git clone <URL-DEL-REPO> restaurante`. Si no, copia la carpeta del proyecto completa a `C:\laragon\www\restaurante`.

7 Instalar dependencias PHP del proyecto

Entra a la carpeta del proyecto en PowerShell: `cd C:\laragon\www\restaurante` y ejecuta `composer install`. Este comando descarga Laravel y todas las librerías (incluido Greenter para SUNAT). Tarda 2-5 minutos según tu internet.

8 Instalar dependencias frontend

En la misma terminal ejecuta `npm install`. Descarga Bootstrap, Vite y las librerías frontend. Después compila los assets para producción con `npm run build`.

9 Crear el archivo de configuración .env

Copia el ejemplo: `copy .env.example .env`. Genera la clave de la app: `php artisan key:generate`. Abre `.env` con tu editor y verifica las credenciales de BD: `DB_DATABASE=restaurante_db`, `DB_USERNAME=root`, `DB_PASSWORD=` (vacío en Laragon por defecto).

10 Crear la base de datos

Abre **phpMyAdmin** desde Laragon (click derecho → MySQL → phpMyAdmin) o desde `http://localhost/phpmyadmin`. Crea una base de datos llamada `restaurante_db` con cotejamiento `utf8mb4_unicode_ci`.

11 Aplicar migraciones y datos iniciales

En PowerShell: `php artisan migrate --seed`. Esto crea todas las tablas y carga usuarios, productos, mesas y configuración de demo. Usuario admin por defecto: `admin@admin.com` · clave: `password`.

12 Crear enlace simbólico de storage

Para que las fotos de productos se sirvan correctamente: `php artisan storage:link`.

13 Arrancar el servidor de desarrollo

Opción A (sencilla): `php artisan serve` y abre `http://localhost:8000`.
Opción B (recomendada en producción local): Laragon detecta automáticamente la carpeta `restaurante` y la expone en `http://restaurante.test` o `http://localhost/restaurante/public`.

14 Iniciar sesión y verificar

Entra al navegador, inicia sesión con `admin@admin.com` / `password`. Cambia la contraseña desde el menú de usuario. Verifica que el dashboard, POS y Configuración carguen correctamente.

✓ **El sistema ya está corriendo localmente.** Si la pantalla de login aparece y puedes entrar como admin, la instalación fue exitosa. Continúa con el Paso 4 (hosting) cuando quieras publicarlo en internet.

4. HOSTING WEB RECOMENDADO

Para una aplicación Laravel hay tres categorías de hosting: **shared hosting** (económico, simple, limitado), **VPS / Cloud** (control total, requiere conocimientos) y **plataformas managed Laravel** (caro pero con todo automatizado). Para un restaurante pequeño/mediano la elección óptima por **precio/facilidad/rendimiento** es Hostinger.

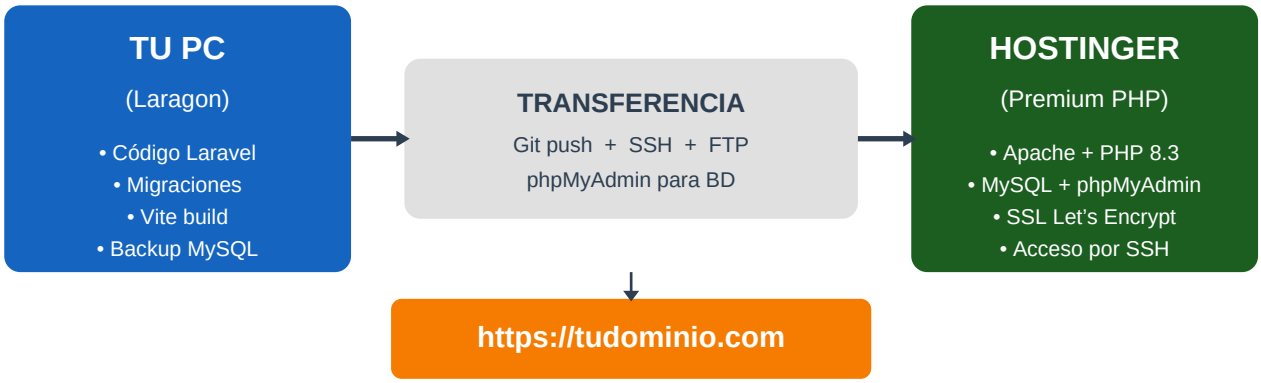
Comparativa de hostings (mayo 2026)

Hosting	Tipo	Precio aprox.	Pros	Contras
Hostinger (★ Recomendado)	Shared / VPS	S/ 11–35 /mes	PHP 8.3 nativo · MySQL · SSH · panel hPanel · SSL gratis · CDN incluido · soporte en español 24/7	Planes más baratos limitan CPU en horas pico
SiteGround	Shared	US$ 7–25 /mes	Excelente rendimiento · soporte 24/7 · WordPress optimizado	Más caro en renovaciones
DigitalOcean	VPS	US$ 6–24 /mes	Control total · escalable · datacenters globales	Requiere admin Linux · sin panel
Laravel Forge + DigitalOcean	Managed VPS	US$ 12+ /mes + servidor	Despliegue automatizado · ideal para Laravel	Costo adicional · curva de aprendizaje
AWS Lightsail	VPS	US$ 5–40 /mes	Marca Amazon · escalable a EC2	Configuración manual · sin soporte local

■ **Recomendación final: Hostinger Premium Web Hosting** (plan ~S/ 14/mes con descuento anual). Razones: incluye PHP 8.3, MySQL, dominio gratis el primer año, SSL Let's Encrypt automático, acceso SSH, soporte en español 24/7 y panel hPanel muy intuitivo. Si tu negocio crece mucho, puedes migrar al plan VPS sin cambiar de proveedor.

5. DESPLIEGUE EN HOSTINGER

DESPLIEGUE LOCAL → PRODUCCIÓN



Estos son los pasos detallados para publicar el sistema en Hostinger. Tiempo total estimado: **1 a 2 horas**. Asegúrate de tener antes: dominio comprado (o usa el subdominio gratuito de Hostinger), las credenciales del hPanel, y el proyecto funcionando localmente.

1 Contratar el plan en Hostinger

Entra a `hostinger.com`. Elige **Premium Web Hosting** (o superior). Si compras 12-48 meses, el primer año el dominio es gratuito (elige `.com`, `.pe`, etc.). Completa el pago.

2 Crear el sitio en hPanel

Ingresa a tu panel Hostinger. Pulsa **Hosting** → **Administrar** sobre tu plan. Selecciona el dominio y pulsa **Crear sitio**. Elige "Sitio en blanco" (no instales WordPress).

3 Asegurar la versión de PHP

En hPanel: **Avanzado** → **Configuración PHP**. Selecciona **PHP 8.3**. En la pestaña **Extensiones**, verifica que estén habilitadas: `soap`, `openssl`, `zip`, `xml`, `mbstring`, `curl`, `intl`, `fileinfo`, `pdo_mysql`, `gd`. Guarda cambios.

4 Crear la base de datos en hPanel

En hPanel: **Bases de datos** → **Bases de datos MySQL**. Crea una nueva con nombre `restaurante_db`, usuario y contraseña fuertes. **Anota estos datos**, los usarás en el `.env` del servidor.

5 Subir el código del proyecto

Tienes 3 opciones (recomendado A si tienes Git en el repo):
A) Git desde hPanel: Avanzado → Git → "Crear repositorio" → pega la URL de tu repo. Hostinger lo clona en `~/domains/tudominio.com/`.
B) ZIP por File Manager: Comprime tu carpeta del proyecto (sin `vendor/` ni `node_modules/`), súbela por File Manager, descomprímela en `domains/tudominio.com/public_html/`.
C) FTP: usa FileZilla con las credenciales FTP del hPanel.

6 Reorganizar carpetas para que apunten a /public

Por seguridad Laravel sirve solo la carpeta `public/`. En Hostinger el directorio público es `public_html`. Tienes dos opciones:
Recomendada: coloca el proyecto en `~/domains/tudominio.com/restaurante/` y haz que `public_html` sea un symlink a `restaurante/public/` (Hostinger ya provee esa opción al crear el sitio: "Directorio raíz del dominio").
Alternativa: en hPanel → Dominios → Configurar → "Carpeta raíz" → cambia a `restaurante/public`.

7 Configurar acceso SSH

En hPanel: **Avanzado** → **Acceso SSH** → habilítalo. Anota el host, puerto, usuario y contraseña. Conéctate desde PowerShell con `ssh -p 65002 usuario@tudominio.com`.

8 Instalar dependencias en el servidor

Conectado por SSH, ve al proyecto: `cd ~/domains/tudominio.com/restaurante`. Ejecuta: `composer install --no-dev --optimize-autoloader`. Esto instala Laravel y Greenter. Si Node está disponible, también `npm install && npm run build` (si no, compila el build localmente y sube la carpeta `public/build/`).

9 Crear el archivo .env de producción

En SSH: `cp .env.example .env`. Edítalo con `nano .env`. Cambia: `APP_ENV=production`, `APP_DEBUG=false`, `APP_URL=https://tudominio.com`, y las credenciales de BD del paso 4. Genera la clave: `php artisan key:generate`.

10 Importar la base de datos

Exporta tu BD local desde phpMyAdmin (Laragon) como SQL. Luego en hPanel: **Bases de datos** → **phpMyAdmin**, selecciona `restaurante_db` y usa **Importar**. Alternativamente, si la BD está vacía: `php artisan migrate --seed --force`.

11 Permisos y enlace de storage

En SSH: `chmod -R 775 storage bootstrap/cache`. Crea el enlace de storage: `php artisan storage:link`. Optimiza cache: `php artisan config:cache`, `php artisan route:cache`, `php artisan view:cache`.

12 Activar SSL (HTTPS)

En hPanel: **Avanzado** → **SSL** → "Instalar SSL gratuito" (Let's Encrypt). Espera 5-10 minutos a que se propague. Activa "Forzar HTTPS" para que toda visita se redirija a la versión segura. Esto es **obligatorio** para que SUNAT acepte tu instalación.

13 Probar el sistema en vivo

Abre `https://tudominio.com`. Inicia sesión como admin. Si todo funciona pero algunos estilos se ven raros, ejecuta una vez más `npm run build` en local y sube la carpeta `public/build/` al servidor por FTP.

14 Configurar facturación electrónica (SUNAT)

Sube tu certificado .pfx desde **/settings** → **Facturación Electrónica**. Cambia el ambiente a PRODUCCIÓN. Realiza una prueba con un comprobante de S/ 1.00 para validar que SUNAT acepta. (Ver guía específica de facturación electrónica.)

15 Configurar tareas programadas (cron)

En hPanel: **Avanzado** → **Cron Jobs**. Crea un cron que ejecute cada minuto: `cd /home/usuario/domains/tudominio.com/restaurante && php artisan schedule:run >> /dev/null 2>&1`. Esto activa los reintentos automáticos de SUNAT y otros jobs programados.

16 Backups automáticos

En hPanel: **Archivos** → **Backups**. Activa los backups diarios (incluidos en el plan Premium). También exporta manualmente cada semana `restaurante_db.sql` desde phpMyAdmin como copia de seguridad adicional.

■ **¡Felicidades!** Tu sistema está en producción. Comparte la URL con tu equipo, crea sus usuarios desde */users* y empieza a operar. Recuerda monitorear */billing* a diario para detectar comprobantes en estado ERROR o REJECTED.

ANEXO · COMANDOS Y URLS ÚTILES

Comandos artisan más usados

Comando	Uso
`php artisan serve`	Arranca servidor de desarrollo en http://localhost:8000
`php artisan migrate`	Aplica migraciones pendientes
`php artisan migrate:fresh --seed`	Recrea TODA la BD desde cero con datos demo (¡borra todo!)
`php artisan db:seed`	Ejecuta todos los seeders
`php artisan storage:link`	Crea symlink para servir archivos de storage/
`php artisan config:cache`	Cachea configuración (acelera prod)
`php artisan config:clear`	Limpia cache de config (en desarrollo)
`php artisan route:list`	Lista todas las rutas registradas
`php artisan make:controller XController`	Crea un controlador
`php artisan make:migration X`	Crea una migración
`php artisan tinker`	REPL interactivo con Eloquent
`npm run dev`	Compila assets en modo desarrollo con hot-reload
`npm run build`	Compila assets optimizados para producción
`composer install`	Instala dependencias PHP
`composer update`	Actualiza dependencias PHP
`composer dump-autoload`	Regenera el autoloader

URLs del sistema una vez instalado

Ruta	Sección
`/login`	Pantalla de inicio de sesión
`/`	Dashboard principal (admin)
`/pos`	Punto de venta (POS)
`/kitchen`	Monitor de cocina
`/sales`	Historial de ventas
`/products`	Gestión de productos
`/categories`	Categorías
`/tables`	Mapa de mesas y salones

`/clients`	Base de clientes
`/reservations`	Reservas
`/delivery`	Delivery
`/expenses`	Gastos
`/cash-registers/open · /close`	Apertura y cierre de caja
`/users`	Usuarios y roles
`/reports`	Reportes BI
`/settings`	Configuración (datos, logo, SUNAT)
`/billing`	Comprobantes electrónicos
`/credit-notes`	Notas de crédito
`/daily-summaries`	Resumen diario de boletas
`/menu`	Menú digital público (cliente final)
`/system`	Reset, backup y restore

Fin del documento. Para profundizar en facturación electrónica, consulta el documento específico *documentacion_facturacion_electronica.pdf*.