



Tecnología, instalación y hosting

Suite Educativa Integral SaaS · Guía completa de puesta en marcha

Tecnologías empleadas y funcionalidades del sistema

Instalación paso a paso en una computadora nueva

Comparativa de hosting y despliegue en producción

Versión 1.0 · Julio 2026

Contenido

- 1 ¿Con qué tecnología está desarrollado el sistema?**
Stack completo · Arquitectura en capas · Por qué se eligió cada pieza
- 2 ¿Qué funcionalidades tiene el sistema?**
Mapa de los 22 módulos · Cadena operativa · Cifras del proyecto
- 3 Instalación en una computadora nueva**
Qué descargar · Instalación de XAMPP, Composer, Git y VS Code
Crear la base de datos · Configurar el proyecto · Arrancar
Solución de problemas frecuentes
- 4 ¿Dónde puedo alojar la aplicación? Comparativa de hosting**
Qué necesita Laravel de un hosting · Las cuatro opciones reales
Comparativa de precios y recomendación
- 5 Despliegue paso a paso en el hosting recomendado**
Contratar · Conectar por SSH · Preparar el servidor
Subir el proyecto · Configurar Nginx · Certificado SSL
Tareas programadas · Copias automáticas · Checklist final
- 6 Mantenimiento y actualizaciones en producción**
- 7 Anexo: comandos de referencia rápida**

1. ¿Con qué tecnología está desarrollado el sistema?

La Suite Educativa Integral es una **aplicación web SaaS multi-institución**. Se ejecuta en un servidor y se accede desde cualquier navegador, sin instalar nada en las computadoras de los usuarios. Funciona igual en Windows, Mac, Linux, tablets y teléfonos.

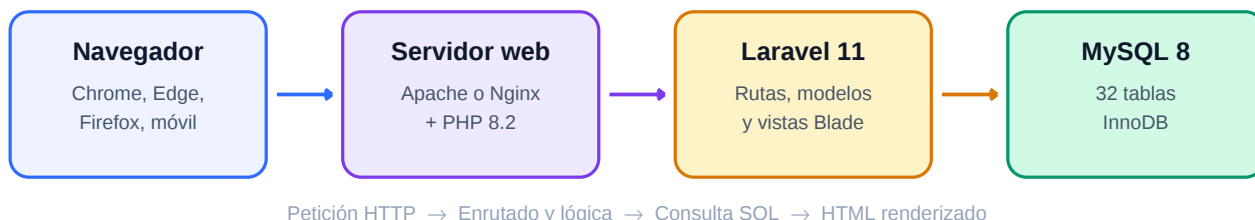


Figura 1. Recorrido de una petición: el navegador pide una página, el servidor web la procesa con PHP, Laravel resuelve la lógica, MySQL entrega los datos y el usuario recibe HTML.

1.1 Stack tecnológico completo

Componente	Tecnología	Versión	Para qué sirve
Lenguaje	PHP	8.2 o superior	Lenguaje de programación del lado del servidor. Ejecuta toda la lógica.
Framework	Laravel	11	Estructura el proyecto: rutas, seguridad, base de datos y plantillas.
Base de datos	MySQL / MariaDB	8.0 / 10.4+	Almacena las 32 tablas con toda la información de las instituciones.
ORM	Eloquent	Incluido en Laravel	Traduce objetos PHP a consultas SQL sin escribir SQL a mano.
Plantillas	Blade	Incluido en Laravel	Motor de vistas. Genera el HTML de las 76 pantallas.
Estilos	Tailwind CSS	3.x por CDN	Diseño visual, colores, espaciados y adaptación a móviles.
Gráficos	Chart.js	4.4.1 por CDN	Los gráficos del dashboard y del módulo de Reportes.
Dependencias	Composer	2.x	Descarga e instala las librerías de PHP que el proyecto necesita.
Servidor web	Apache o Nginx	2.4 / 1.18+	Recibe las peticiones del navegador y se las entrega a PHP.
Versionado	Git	2.x	Control de versiones del código. Opcional pero muy recomendable.

1.2 Arquitectura en capas

El sistema sigue el patrón **Modelo–Vista–Controlador**. Cada capa tiene una responsabilidad clara y sólo se comunica con la capa contigua. Esto permite modificar el diseño sin tocar la lógica, o cambiar la lógica sin

rehacer las pantallas.

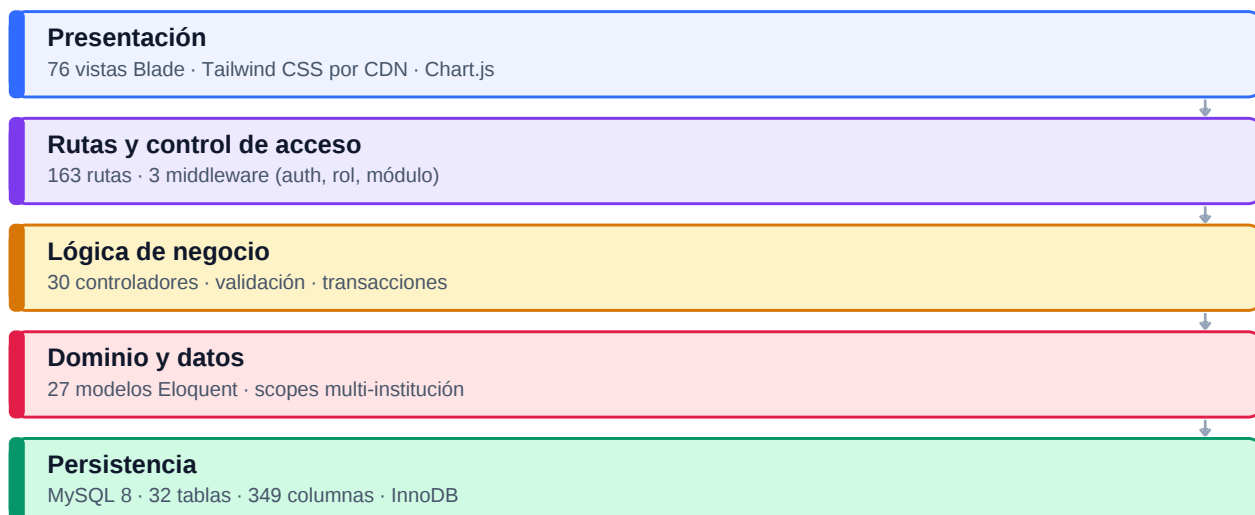


Figura 2. Las cinco capas del sistema, con las cifras reales del proyecto en cada nivel.

1.3 Arquitectura multi-institución

Ésta es la característica que convierte al sistema en un **SaaS** y no en un programa para una sola escuela. Una única instalación atiende a muchas instituciones y cada una ve exclusivamente sus datos.

Una instalación · Una base de datos · Datos aislados por institución

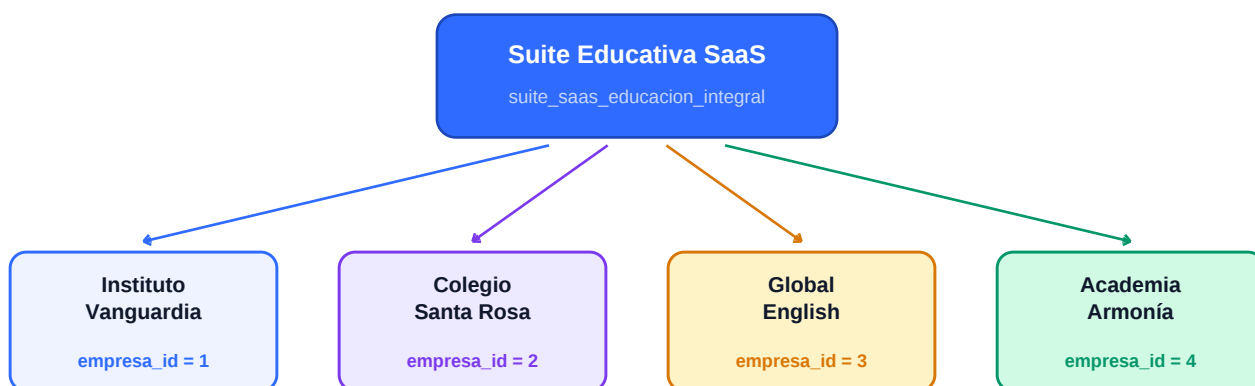


Figura 3. Todas las instituciones comparten la instalación y la base, pero cada registro lleva marcada la columna `empresa_id`, que actúa como frontera infranqueable entre clientes.

Por qué esto importa comercialmente

Con una sola instalación puedes vender a decenas de instituciones. No mantienes un sistema por cliente: actualizas una vez y todos reciben la mejora. El costo de infraestructura por cliente adicional es prácticamente cero, y ahí está el margen del negocio SaaS.

1.4 Por qué se eligió cada tecnología

Decisión	Motivo
PHP y Laravel	PHP se ejecuta en prácticamente cualquier hosting del mundo, incluidos los económicos. Laravel aporta seguridad, estructura y una comunidad enorme en español.
MySQL	Es la base de datos más disponible en hosting. Cualquier proveedor la incluye sin costo adicional.
Tailwind por CDN y no compilado	Elimina la necesidad de Node.js y del proceso de compilación. El sistema se despliega copiando archivos, sin cadena de construcción. Simplifica enormemente el despliegue.
Chart.js en lugar de una librería de pago	Es gratuita, ligera y sus gráficos se adaptan solos al tamaño de la pantalla.
Sin framework de JavaScript	No usa React ni Vue. Las pantallas se generan en el servidor, lo que reduce la complejidad, mejora el posicionamiento y hace el sistema mucho más fácil de mantener.

2. ¿Qué funcionalidades tiene el sistema?

El sistema cubre la operación completa de una institución educativa, desde que alguien pregunta por un curso hasta que egresa con su certificado. Está organizado en **22 módulos** agrupados en cuatro áreas funcionales.



Figura 4. Mapa completo de módulos. Cada institución ve sólo los que su plan incluye y el administrador de la plataforma activó.

2.1 La cadena operativa

Los módulos no son islas: forman una cadena donde cada eslabón alimenta al siguiente. Un dato se registra una sola vez y viaja por todo el sistema.

Cadena operativa del sistema



Figura 5. Recorrido de un estudiante por el sistema, del primer contacto comercial al certificado final.

2.2 Funcionalidades por área

Plataforma SaaS (Super Administrador)

- **Gestión de instituciones** — alta, edición, suspensión y eliminación de empresas cliente, con asignación de tipo de negocio entre 9 categorías educativas.
- **Planes de suscripción** — creación de paquetes comerciales con precio mensual y anual, límites de estudiantes y usuarios, y selección de módulos incluidos.

- **Suscripciones** — registro de contratos con cálculo automático de precio y fechas, historial completo por cliente y emisión de ticket.
- **Módulos visibles** — interruptor por institución para encender o apagar cada módulo.
- **Mantenimiento de plataforma** — copias de seguridad totales o por institución, restauración y reseteo selectivo.

Área académica

- **CRM / Prospectos** — embudo de captación con estados, canal de origen, agenda de seguimiento y conversión directa a estudiante.
- **Admisiones** — matrícula con generación automática del cargo de matrícula y hasta 12 pensiones; edición, anulación con motivo y reactivación.
- **Estudiantes** — ficha completa con datos del apoderado, seis estados, búsqueda y filtros.
- **Docentes** — plana docente con profesión, especialidad y tipo de contrato.
- **Programas y Cursos** — catálogo académico con seis tipos (carrera, curso, taller, idioma, programa, diplomado), tres modalidades, cupo y precio.
- **Aulas y Ambientes** — espacios físicos y virtuales con capacidad y equipamiento.
- **Horarios** — programación semanal cruzando programa, docente, aula, día y rango horario.
- **Asistencia** — registro diario por clase con cuatro estados.
- **Evaluaciones** — exámenes con nota máxima y peso, pantalla de calificación masiva y cálculo automático de promedios y tasa de aprobación.
- **Certificados** — cinco tipos de documento con código único y versión imprimible.
- **Campus Virtual** — publicación de material, videos, enlaces y tareas por curso.
- **Biblioteca** — catálogo con ejemplares y control de préstamos y devoluciones.

Área financiera

- **Pagos y Cobranzas** — seis conceptos, cuatro métodos de pago, soporte de **pagos parciales** con cálculo automático de saldo y detección de cuotas vencidas.
- **Caja** — apertura por turno, registro de ingresos y egresos, y cierre con arqueo que compara el monto esperado contra el contado.

Área administrativa

- **Comunicaciones** — anuncios segmentados por audiencia con cálculo de alcance.
- **Recursos Humanos** — personal completo por áreas y contratos, con cálculo de masa salarial y antigüedad.
- **Inventario** — activos, materiales y consumibles con kardex de movimientos y alerta de stock mínimo.
- **Reportes** — indicadores, cuatro gráficos, **exportación a Excel en cinco conjuntos de datos** y versión imprimible membretada.
- **Configuración** — datos de la institución, usuarios y roles.
- **Mantenimiento** — copia, restauración y reseteo de los datos propios de la institución.

2.3 Características transversales

Característica	Detalle
Multi-institución	Aislamiento total de datos con triple capa de protección.

Característica	Detalle
Control de acceso	Seis roles: superadmin, admin, coordinador, docente, secretaria y estudiante.
Diseño adaptable	Todas las pantallas funcionan en computadora, tablet y teléfono.
Identidad por cliente	Cada institución define su logo, color primario, moneda y símbolo.
Copias de seguridad	Generación, descarga y restauración desde el navegador, sin acceso al servidor.
Impresión y exportación	Certificados, recibos y reportes con formato de impresión y salida a Excel.

2.4 El sistema en cifras

Elemento	Cantidad	Elemento	Cantidad
Módulos	22	Tablas de base de datos	32
Rutas	163	Columnas documentadas	349
Controladores	30	Modelos de datos	27
Vistas	76	Tipos de negocio soportados	9

3. Instalación en una computadora nueva

Esta sección asume una computadora con Windows recién formateada, sin nada instalado. Al terminar tendrás el sistema funcionando localmente en tu navegador. **Tiempo estimado: 40 a 60 minutos.**

XAMPP (un solo instalador)

- Apache** servidor web
- PHP 8.2** intérprete
- MySQL / MariaDB** base de datos
- phpMyAdmin** gestor visual de la BD

Se instalan aparte

- Composer** gestor de paquetes PHP
- Git** control de versiones
- VS Code** editor de código

Resultado: `php artisan serve --port=8062`

El sistema queda disponible en `http://127.0.0.1:8062`

Figura 6. Todo lo que necesitas: XAMPP agrupa el servidor web, PHP y MySQL en un solo instalador. Composer, Git y el editor se instalan por separado.

3.1 Qué descargar (los cuatro programas)

Programa	Dónde descargarlo	Qué versión	Tamaño
XAMPP	apachefriends.org	La que incluya PHP 8.2 o superior	~150 MB
Composer	getcomposer.org/download	Composer-Setup.exe (Windows)	~2 MB
Git	git-scm.com/downloads	Última estable de 64 bits	~60 MB
Visual Studio Code	code.visualstudio.com	Última estable	~90 MB

Descarga siempre del sitio oficial

No descargues estos programas de sitios de terceros ni de enlaces de foros. XAMPP y Composer tienen acceso completo a tu sistema; una versión modificada es un riesgo serio de seguridad.

3.2 Paso 1 — Instalar XAMPP

XAMPP es un paquete que instala de una sola vez el servidor web Apache, el intérprete de PHP y la base de datos MySQL. Es la forma más rápida de tener el entorno completo.

1 Ejecuta el instalador

Haz clic derecho sobre el archivo descargado y elige **Ejecutar como administrador**. Si Windows muestra una advertencia de control de cuentas, acepta.

2

Selecciona los componentes

Deja marcados como mínimo **Apache**, **MySQL**, **PHP** y **phpMyAdmin**. Puedes desmarcar Perl, Tomcat, FileZilla y Mercury: no los necesitas.

3

Elige la carpeta de instalación

Acepta la ruta por defecto `C:\xampp`. Evita instalarlo dentro de **Archivos de programa**, porque los permisos de Windows dan problemas.

4

Termina y abre el Panel de Control

Al finalizar, marca la opción de abrir el panel. Es la ventana desde donde encenderás los servicios.

5

Enciende Apache y MySQL

En el panel, presiona **Start** en la fila de Apache y en la de MySQL. Ambas deben quedar resaltadas en verde.

6

Verifica que funciona

Abre el navegador y entra a `http://localhost`. Debe aparecer la página de bienvenida de XAMPP. Si aparece, el servidor está operativo.

Si Apache no arranca: el conflicto del puerto 80

El error más común. Skype, IIS o algún antivirus suelen ocupar el puerto 80. Solución: en el Panel de XAMPP presiona **Config** en la fila de Apache → `httpd.conf`, busca **Listen 80** y cámbialo por **Listen 8080**. Guarda y vuelve a presionar Start. Ahora entrarás a `http://localhost:8080`. Para nuestro caso esto no es crítico, porque usaremos el servidor propio de Laravel.

3.3 Paso 2 — Verificar la versión de PHP

Laravel 11 exige PHP 8.2 como mínimo. Comprobémoslo antes de continuar.

Símbolo del sistema (CMD) o PowerShell

```
# Añade PHP al PATH temporalmente y consulta la versión
```

```
$ C:\xampp\php\php.exe -v
```

```
PHP 8.2.12 (cli) (built: Oct 24 2025 21:15:22)
```

```
Copyright (c) The PHP Group
```

```
Zend Engine v4.2.12, Copyright (c) Zend Technologies
```

Añadir PHP al PATH de Windows (recomendado)

Para no escribir la ruta completa cada vez: **Menú Inicio** → **"variables de entorno"** → **Variables de entorno** → **Path** → **Editar** → **Nuevo** y agrega `C:\xampp\php`. Cierra y vuelve a abrir la terminal. Ahora basta con escribir `php -v`.

Extensiones de PHP requeridas

XAMPP ya incluye casi todas activadas. Verifica que en `C:\xampp\php\php.ini` estas líneas **no** tengan punto y coma al inicio:

```
C:\xampp\php\php.ini
```

```
extension=pdo_mysql  
extension=mbstring  
extension=openssl  
extension=fileinfo  
extension=curl  
extension=zip  
extension=gd
```

Si alguna tiene ; delante, quítalo, guarda el archivo y reinicia Apache desde el panel.

3.4 Paso 3 — Instalar Composer

Composer descarga las librerías que Laravel necesita. Sin él, el proyecto no arranca.

1 Ejecuta Composer-Setup.exe

Elige la instalación para todos los usuarios si te lo pregunta.

2 Indica la ruta de PHP

El instalador pedirá el ejecutable de PHP. Señala `C:\xampp\php\php.exe`. Normalmente lo detecta solo.

3 Deja en blanco el proxy

Salvo que tu red corporativa use uno.

4 Finaliza y verifica

Cierra todas las terminales abiertas, abre una nueva y ejecuta el comando de verificación.

Verificación de Composer

```
$ composer --version
```

```
Composer version 2.7.6 2026-05-04 14:22:11
```

3.5 Paso 4 — Instalar Git y Visual Studio Code

Git te permite versionar el código y, más adelante, desplegarlo al servidor. VS Code es el editor donde harás cualquier ajuste.

- **Git** — acepta todas las opciones por defecto del instalador. La única que conviene revisar es el editor por defecto: elige Visual Studio Code si ya lo instalaste.
- **VS Code** — instálalo y, dentro, agrega las extensiones **PHP Intelephense** y **Laravel Blade Snippets**. No son obligatorias pero te ahorrarán muchos errores.

3.6 Paso 5 — Colocar el proyecto

Copia la carpeta del sistema a una ruta sencilla y sin espacios ni tildes. Una buena opción es `C:\proyectos\suite-educativa`.

Si el proyecto está en un repositorio Git

```
# Ubícate donde quieras guardarlo
```

```
$ cd C:\proyectos
```

```
# Clona el repositorio
```

```
$ git clone https://github.com/tu-usuario/suite-educativa.git
```

```
$ cd suite-educativa
```

3.7 Paso 6 — Instalar las dependencias del proyecto

Este comando lee el archivo `composer.json` y descarga Laravel y todas sus librerías dentro de la carpeta `vendor\`. Tarda entre 2 y 5 minutos según tu conexión.

Dentro de la carpeta del proyecto

```
$ composer install
```

```
Installing dependencies from lock file (including require-dev)
```

```
Verifying lock file contents can be installed on current platform.
```

```
Package operations: 112 installs, 0 updates, 0 removals
```

```
- Downloading laravel/framework (v11.x)
```

```
...
```

```
Generating optimized autoload files
```

```
@php artisan package:discover --ansi
```

Si Composer falla por certificados SSL

En algunas instalaciones de XAMPP falta el archivo de certificados. Descarga `cacert.pem` desde curl.se/ca/cacert.pem, guárdalo en `C:\xampp\php\extras\ssl\` y añade en `php.ini`: `curl.cainfo = "C:\xampp\php\extras\ssl\cacert.pem"`. Reinicia la terminal.

3.8 Paso 7 — Crear la base de datos

Tienes dos formas. La visual con phpMyAdmin es la más cómoda si estás empezando.

Opción A — Con phpMyAdmin (visual)

- 1 Abre phpMyAdmin**
Entra a <http://localhost/phpmyadmin> en el navegador.
- 2 Ve a la pestaña "Bases de datos"**
Está en el menú superior.
- 3 Escribe el nombre**
Exactamente `suite_saas_educacion_integral`, sin mayúsculas ni espacios.
- 4 Elige el cotejamiento**
Selecciona `utf8mb4_unicode_ci`. Esto es lo que permite guardar tildes y la letra ñ correctamente.
- 5 Presiona "Crear"**
La base aparecerá vacía en el panel izquierdo. Es lo esperado.

Opción B — Por línea de comandos

Consola de MySQL

```
$ C:\xampp\mysql\bin\mysql.exe -u root

CREATE DATABASE suite_saas_educacion_integral
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;

SHOW DATABASES;

EXIT;
```

3.9 Paso 8 — Configurar el archivo .env

El archivo `.env` guarda la configuración del entorno: cómo conectarse a la base de datos, en qué modo funciona la aplicación y su clave de cifrado.

- 1 Copia el archivo de ejemplo**
En la carpeta del proyecto verás `.env.example`. Duplícalo y renombra la copia a `.env` (con el punto delante y sin extensión).
- 2 Ábrelo con VS Code**
Y ajusta el bloque de base de datos como se muestra abajo.
- 3 Genera la clave de la aplicación**
Es obligatoria: cifra las sesiones y las contraseñas.

Contenido del archivo .env

```
APP_NAME="Suite Educativa Integral"
APP_ENV=local
APP_KEY=
APP_DEBUG=true
APP_URL=http://127.0.0.1:8062

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=suite_saas_educacion_integral
DB_USERNAME=root
DB_PASSWORD=
```

Generar la clave

```
$ php artisan key:generate
```

```
INFO Application key set successfully.
```

La contraseña de root en XAMPP

Por defecto XAMPP deja la contraseña de **root** vacía, por eso **DB_PASSWORD=** va sin valor. Si tú le pusiste una, escríbela ahí. En un servidor de producción **jamás** uses root sin contraseña.

3.10 Paso 9 — Crear las tablas y cargar los datos

Las migraciones construyen las 32 tablas. Los seeders cargan los catálogos (tipos de negocio, planes, módulos) y datos de demostración para que puedas probar el sistema de inmediato.

Construcción de la base de datos

```
# Crea las 32 tablas
```

```
$ php artisan migrate
```

```
INFO Running migrations.
```

```
0001_01_01_000001_create_users_table ..... DONE
```

```
0001_01_01_000002_create_empresas_table ..... DONE
```

```
0001_01_01_000003_create_modulos_table ..... DONE
```

```
...
```

```
# Carga catálogos y datos de ejemplo
```

```
$ php artisan db:seed
```

```
INFO Seeding database.
```

```
Database seeding completed successfully.
```

3.11 Paso 10 — Arrancar el sistema

Levantar el servidor de desarrollo

```
$ php artisan serve --port=8062
```

```
INFO Server running on [http://127.0.0.1:8062].
```

Press Ctrl+C to stop the server

Abre el navegador en <http://127.0.0.1:8062>. Verás la pantalla de acceso. Ingresa con las credenciales de demostración:

Perfil	Correo	Contraseña	A dónde entra
Super Admin SaaS	superadmin@suite.test	password	Panel de la plataforma
Admin institución	admin@globalenglish.edu.pe	password	Panel de la institución

¡Listo! El sistema ya funciona en tu computadora

Mientras la ventana de la terminal permanezca abierta, el sistema estará disponible. Al cerrarla se detiene. Para volver a usarlo: enciende MySQL en XAMPP y ejecuta de nuevo `php artisan serve --port=8062`.

3.12 Solución de problemas frecuentes

Mensaje o síntoma	Causa	Solución
SQLSTATE[HY000] [2002] No se puede establecer conexión	MySQL no está encendido.	Abre el Panel de XAMPP y presiona Start en la fila de MySQL.
SQLSTATE[HY000] [1049] Unknown database	La base no existe o el nombre no coincide.	Créala en phpMyAdmin con el nombre exacto y revisa <code>DB_DATABASE</code> en el <code>.env</code> .
No application encryption key has been specified	Falta la clave de la aplicación.	Ejecuta <code>php artisan key:generate</code> .
Class "PDO" not found	La extensión <code>pdo_mysql</code> está desactivada.	Quita el <code>;</code> de <code>extension=pdo_mysql</code> en <code>php.ini</code> y reinicia.
La página se ve sin estilos, todo en blanco y negro	No hay conexión a internet: Tailwind se carga por CDN.	Conecta a internet. En desarrollo sin red, considera descargar Tailwind localmente.
Los cambios en las vistas no se reflejan	Caché de vistas compiladas.	Ejecuta <code>php artisan view:clear</code> .
'composer' no se reconoce como comando	Composer no quedó en el PATH.	Cierra y abre la terminal. Si persiste, reinstala Composer.
Página en blanco sin ningún mensaje	Error de PHP con los errores ocultos.	Pon <code>APP_DEBUG=true</code> en el <code>.env</code> y revisa <code>storage/logs/laravel.log</code> .

4. ¿Dónde alojar la aplicación? Comparativa de hosting

Aquí viene la decisión más importante del proyecto. Un hosting mal elegido te obligará a migrar todo en pocos meses, así que conviene entender primero qué necesita realmente el sistema.

4.1 Qué exige Laravel de un hosting

Requisito	Por qué es indispensable
PHP 8.2 o superior	Laravel 11 no arranca con versiones anteriores.
Acceso SSH (terminal)	Sin terminal no puedes ejecutar <code>composer install</code> ni <code>php artisan migrate</code> . Es el requisito que descarta a la mayoría de hostings baratos.
Composer disponible	Instala las dependencias del proyecto.
Raíz configurable	El dominio debe apuntar a la carpeta <code>public/</code> , no a la raíz del proyecto. Si el proveedor no lo permite, expones tu archivo <code>.env</code> con las contraseñas.
Base de datos MySQL 8	Con permisos para crear tablas e índices.
Tareas programadas (cron)	Para copias automáticas y procesos periódicos.
Certificado SSL	Obligatorio: manejas datos personales de menores de edad.

El punto que decide todo: acceso SSH

Muchos hostings compartidos económicos no dan terminal ni permiten cambiar la raíz del dominio. En esos servidores **no puedes desplegar Laravel correctamente**. Existen trucos para forzarlo, pero comprometen la seguridad y te darán problemas en cada actualización. Es la razón principal por la que aquí se recomienda un VPS.

4.2 Las cuatro opciones reales

Opción	Cómo funciona	Ventaja	Desventaja
Hosting compartido	Compartes un servidor con cientos de sitios. Panel tipo cPanel.	El más barato. No requiere conocimientos de servidor.	Muchos no dan SSH. Recursos limitados. Difícil de escalar.
VPS (servidor virtual privado)	Un servidor virtual sólo tuyo, con acceso total de administrador.	Control completo, buen precio, escala cuando creces.	Debes configurarlo tú la primera vez y encargarte de las actualizaciones.
Hosting gestionado para Laravel	Servicios como Laravel Forge o Laravel Cloud que automatizan el servidor.	Despliegue en minutos, sin tocar configuración de servidor.	Cuesta más: pagas el servicio y además el servidor.
Nube (AWS, Google Cloud)	Infraestructura empresarial por componentes.	Escala prácticamente sin límite.	Complejidad y costos altos. Desproporcionado para empezar.

4.3 Comparativa de precios

Costo mensual aproximado en dólares

Referencial · julio 2026 · contratación a 24 meses



Figura 7. Costo mensual aproximado. Precios referenciales de julio de 2026 en contratación a 24 meses; verifica siempre en el sitio del proveedor antes de contratar, porque cambian con frecuencia.

Proveedor y plan	Recursos	Precio aprox.	Apto para este sistema
Hostinger VPS KVM 1	1 vCPU · 4 GB RAM · 50 GB NVMe	\$ 6.49 / mes	Sí, para arrancar con 1 a 3 instituciones.
Hostinger VPS KVM 2	2 vCPU · 8 GB RAM · 100 GB NVMe	\$ 8.99 / mes aprox.	Sí. Es la opción recomendada.
Hostinger VPS KVM 8	8 vCPU · 32 GB RAM · 400 GB NVMe	\$ 25.99 / mes	Sí, cuando superes las 20 instituciones activas.
DigitalOcean / Vultr	Desde 1 vCPU · 1–2 GB RAM	Desde \$ 5–6 / mes	Sí, pero el panel y el soporte están sólo en inglés.
SiteGround GrowBig	Compartido · 20 GB SSD	\$ 6.69 / mes	Limitado. Sirve para una sola institución pequeña.
Laravel Forge	Servicio de gestión (no incluye servidor)	Desde \$ 12 / mes + servidor	Sí, si prefieres no administrar el servidor tú mismo.
Laravel Cloud	Plataforma totalmente gestionada, pago por uso	Variable · \$5 de crédito inicial	Sí, pero el costo es menos predecible al crecer.

4.4 Recomendación

Recomendación: Hostinger VPS, plan KVM 2

Para un negocio SaaS que empieza a vender a institutos y academias, es el mejor equilibrio entre precio, capacidad y facilidad. Las razones concretas:

1. Acceso SSH y root completo — puedes ejecutar Composer y Artisan sin restricciones.
2. Instalación de Laravel con un clic desde su panel, que deja PHP, MySQL y Nginx listos.
3. Panel en español y soporte en español las 24 horas, algo que DigitalOcean y Vultr no ofrecen.
4. Con 8 GB de RAM soportas cómodamente entre 15 y 25 instituciones activas.
5. Certificado SSL gratuito, copias automáticas y cortafuegos incluidos.
6. Garantía de devolución de 30 días si no te convence.

Cuándo elegir otra opción

Si sólo vas a atender a **una institución pequeña** y no piensas revender el sistema, un hosting compartido con SSH (como SiteGround GrowBig) te alcanza y ahorra configuración. Si **no quieres administrar servidores en absoluto** y el presupuesto no es problema, Laravel Forge sobre un servidor de Hetzner o DigitalOcean automatiza todo el mantenimiento.

5. Despliegue paso a paso en el hosting recomendado

A continuación, el procedimiento completo para publicar el sistema en un VPS de Hostinger con Ubuntu. Los mismos pasos sirven, con mínimas variaciones, para cualquier VPS con Ubuntu.

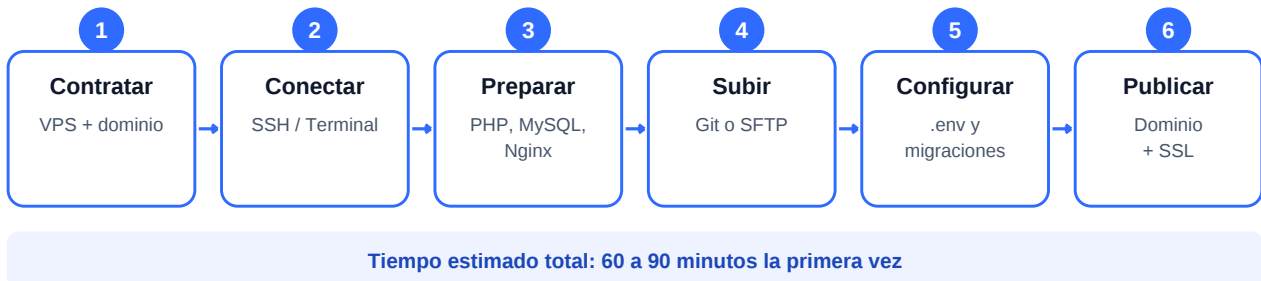


Figura 8. Las seis etapas del despliegue.

5.1 Etapa 1 — Contratar el VPS y el dominio

1 Contrata el VPS

En [hostinger.com](#) elige **VPS Hosting** → **plan KVM 2**. El pago a 24 meses tiene el mejor precio, pero puedes empezar con un plazo corto para probar.

2 Elige el sistema operativo con Laravel preinstalado

Durante la configuración inicial, en **Sistema operativo** busca la opción **Ubuntu 24.04 con Laravel** (en el apartado de aplicaciones). Esto te ahorra instalar PHP, MySQL y Nginx manualmente.

3 Define la contraseña de root

Usa una contraseña larga y guárdala en un gestor. Es la llave maestra del servidor.

4 Anota la dirección IP

Al terminar el aprovisionamiento verás algo como `82.180.xxx.xxx`. La necesitarás enseguida.

5 Contrata o apunta tu dominio

Si compras el dominio en Hostinger, la conexión es automática. Si lo tienes en otro proveedor, ve a su panel de DNS y crea un registro **A** apuntando a la IP del VPS.

Registro DNS que debes crear

En el panel DNS de tu dominio

Tipo	Nombre	Valor	TTL
A	@	82.180.xxx.xxx	3600
A	www	82.180.xxx.xxx	3600

La propagación tarda entre 15 minutos y 24 horas

5.2 Etapa 2 — Conectarte al servidor por SSH

Desde Windows 10 en adelante no necesitas instalar nada: PowerShell ya incluye SSH.

PowerShell en tu computadora

```
# Conéctate al servidor (reemplaza por tu IP real)
$ ssh root@82.180.xxx.xxx

The authenticity of host '82.180.xxx.xxx' can't be established.
Are you sure you want to continue connecting (yes/no)? yes

root@82.180.xxx.xxx's password: (escribe tu contraseña, no se ve al teclear)

Welcome to Ubuntu 24.04.1 LTS
root@srv123456:~#
```

La contraseña no se ve mientras la escribes

Es normal y es a propósito: Linux no muestra asteriscos. Escríbela completa y presiona Enter.

5.3 Etapa 3 — Preparar el servidor

En el servidor (por SSH)

```
# 1. Actualiza el sistema
$ apt update && apt upgrade -y

# 2. Verifica que PHP esté instalado y en la versión correcta
$ php -v

# 3. Instala las extensiones que Laravel necesita
$ apt install -y php8.2-mbstring php8.2-xml php8.2-curl \
php8.2-zip php8.2-mysql php8.2-gd unzip git

# 4. Verifica Composer (si no está, instálalo)
$ composer --version

# 5. Verifica Nginx y MySQL
$ systemctl status nginx
$ systemctl status mysql
```

Crear la base de datos y su usuario

En producción **no se usa root**. Crearemos un usuario dedicado con permisos sólo sobre nuestra base.

Consola de MySQL en el servidor

```
$ mysql -u root -p

CREATE DATABASE suite_saas_educacion_integral
CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

CREATE USER 'suite_user'@'localhost'
IDENTIFIED BY 'UnaClaveLargaYSegura2026!';

GRANT ALL PRIVILEGES ON suite_saas_educacion_integral.*
TO 'suite_user'@'localhost';

FLUSH PRIVILEGES;

EXIT;
```

5.4 Etapa 4 — Subir el proyecto

Dos caminos. El primero es el recomendado porque facilita las actualizaciones futuras.

Opción A — Con Git (recomendado)

En el servidor

```
# Ubícate en la carpeta web
$ cd /var/www

# Clona tu repositorio
$ git clone https://github.com/tu-usuario/suite-educativa.git

$ cd suite-educativa
```

Opción B — Con SFTP (sin repositorio)

1

Descarga FileZilla

Desde filezilla-project.org, versión cliente.

2

Conéctate

Servidor: `sftp://82.180.xxx.xxx` · Usuario: `root` · Puerto: `22`.

3

Sube el proyecto

Arrastra la carpeta a `/var/www/`. **No subas** las carpetas `vendor/` ni `node_modules/`: se generan en el servidor y pesan mucho.

5.5 Etapa 5 — Configurar el proyecto

En `/var/www/suite-educativa`

```
# 1. Instala las dependencias en modo producción
$ composer install --optimize-autoloader --no-dev

# 2. Crea el archivo de configuración
$ cp .env.example .env
$ nano .env
```

Dentro del editor `nano`, ajusta estos valores:

Archivo `.env` para PRODUCCIÓN

```
APP_NAME="Suite Educativa Integral"
APP_ENV=production
APP_KEY=
APP_DEBUG=false
APP_URL=https://suite.tudominio.com
```

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=suite_saas_educacion_integral
DB_USERNAME=suite_user
DB_PASSWORD=UnaClaveLargaYSegura2026!
```

Los dos valores que jamás debes equivocar en producción

`APP_ENV=production` y `APP_DEBUG=false`. Si dejas el modo depuración activo, cualquier error mostrará al visitante las rutas de tu servidor, fragmentos de código y hasta las credenciales de la base de datos. Es la falla de seguridad más común y más grave en despliegues de Laravel.

Guarda con `Ctrl+O`, Enter, y sal con `Ctrl+X`. Después:

```
Terminar la configuración

# 3. Genera la clave de la aplicación
$ php artisan key:generate

# 4. Crea las tablas
$ php artisan migrate --force

# 5. Carga los catálogos (tipos de negocio, planes, módulos)
$ php artisan db:seed --class=TipoNegocioSeeder
$ php artisan db:seed --class=PlanSeeder
$ php artisan db:seed --class=ModuloSeeder

# 6. Optimiza para producción
$ php artisan config:cache
$ php artisan route:cache
$ php artisan view:cache
```

No cargues los seeders de demostración en producción

Los seeders de estudiantes, docentes, pagos y empresas demo son datos ficticios. En producción carga **sólo** los tres catálogos indicados arriba. Las instituciones reales las crearás desde el panel del Super Admin.

Permisos de carpetas

```
Permisos correctos

# El servidor web debe poder escribir en storage y bootstrap/cache
$ chown -R www-data:www-data /var/www/suite-educativa
$ chmod -R 775 /var/www/suite-educativa/storage
$ chmod -R 775 /var/www/suite-educativa/bootstrap/cache

# Crea la carpeta de copias de seguridad
```

```
$ mkdir -p /var/www/suite-educativa/storage/app/backups  
$ chown -R www-data:www-data /var/www/suite-educativa/storage/app/backups
```

5.6 Etapa 6 — Configurar el dominio en Nginx

Estructura en el servidor

/var/www/suite-educativa/

- app/ bootstrap/ config/
- database/ resources/ routes/
- storage/ ← permisos 775
- vendor/ ← composer install
- .env ← configuración
- public/ ← RAÍZ DEL SITIO
- index.php

Configuración de Nginx

```
server {  
    server_name suite.tudominio.com;  
    root /var/www/suite-educativa/public;  
  
    index index.php;  
    charset utf-8;  
  
    location / {  
        try_files $uri $uri/ /index.php?$query_string;  
    }  
}
```

La raíz apunta a public/, nunca a la carpeta del proyecto

Figura 9. Estructura del proyecto en el servidor y configuración mínima de Nginx. Nota que la raíz apunta a public/, nunca a la carpeta del proyecto.

Crear el archivo de configuración del sitio

```
$ nano /etc/nginx/sites-available/suite-educativa
```

Contenido completo del archivo

```
server {  
    listen 80;  
  
    server_name suite.tudominio.com www.suite.tudominio.com;  
    root /var/www/suite-educativa/public;  
  
    add_header X-Frame-Options "SAMEORIGIN";  
    add_header X-Content-Type-Options "nosniff";  
  
    index index.php;  
    charset utf-8;  
    client_max_body_size 64M;  
  
    location / {  
        try_files $uri $uri/ /index.php?$query_string;  
    }  
  
    location ~ \.php$ {  
        fastcgi_pass unix:/var/run/php/php8.2-fpm.sock;  
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;  
        include fastcgi_params;  
    }  
  
    location ~ /\.(!well-known).* {  
        deny all;  
    }  
}
```

```
}  
}
```

Activar el sitio

```
# Crea el enlace simbólico  
$ ln -s /etc/nginx/sites-available/suite-educativa \  
/etc/nginx/sites-enabled/  
  
# Desactiva el sitio por defecto  
$ rm /etc/nginx/sites-enabled/default  
  
# Verifica que la configuración sea válida  
$ nginx -t  
  
nginx: configuration file /etc/nginx/nginx.conf test is successful  
  
# Recarga Nginx  
$ systemctl reload nginx
```

5.7 Certificado SSL gratuito

Sin HTTPS el navegador marcará tu sitio como "No seguro" y estarás transmitiendo datos personales de menores de edad sin cifrar. Let's Encrypt lo resuelve gratis en dos minutos.

Instalar el certificado

```
# Instala Certbot  
$ apt install -y certbot python3-certbot-nginx  
  
# Solicita y configura el certificado automáticamente  
$ certbot --nginx -d suite.tudominio.com -d www.suite.tudominio.com  
  
# Responde: tu correo, acepta los términos,  
# y elige la opción 2 (redirigir todo el tráfico a HTTPS)  
  
# Verifica que la renovación automática funcione  
$ certbot renew --dry-run
```

El certificado se renueva solo

Let's Encrypt emite certificados de 90 días, pero Certbot instala una tarea programada que los renueva automáticamente. No tienes que hacer nada más.

5.8 Tareas programadas y copias automáticas

Configurar el programador de Laravel

```
$ crontab -e
```

Agrega esta línea al final del archivo:

```
* * * * * cd /var/www/suite-educativa && php artisan schedule:run >> /dev/null 2>&1
```

Copia de seguridad automática de la base de datos

Crea el script

```
$ nano /root/backup-suite.sh
```

```
#!/bin/bash
```

```
FECHA=$(date +%Y%m%d_%H%M%S)
```

```
mysqldump -u suite_user -p'UnaClaveLargaYSegura2026!' \
```

```
suite_saas_educacion_integral > /root/backups/suite_$FECHA.sql
```

Conserva sólo los últimos 14 días

```
find /root/backups -name "suite_*.sql" -mtime +14 -delete
```

Dale permisos de ejecución

```
$ mkdir -p /root/backups && chmod +x /root/backup-suite.sh
```

Prográmalo todos los días a las 3 de la madrugada

```
$ crontab -e
```

```
0 3 * * * /root/backup-suite.sh
```

Una copia en el mismo servidor no es una copia de seguridad

Si el servidor falla, pierdes el sistema y los respaldos. Configura además la copia automática que ofrece Hostinger en su panel (VPS → Backups), o descarga periódicamente los archivos a tu computadora o a un almacenamiento en la nube.

5.9 Checklist final antes de entregar a clientes

#	Verificación	Cómo comprobarlo
1	El sitio abre por HTTPS con candado	Entra a https://suite.tudominio.com y revisa el candado del navegador.
2	<code>APP_DEBUG=false</code>	Provoca un error a propósito: debe mostrar una página genérica, nunca código.
3	El archivo <code>.env</code> no es accesible	Entra a https://suite.tudominio.com/.env . Debe dar error 404.
4	La raíz apunta a <code>public/</code>	Entra a https://suite.tudominio.com/storage/logs/laravel.log . Debe dar error, no mostrar el registro.
5	El acceso funciona	Ingresa con el usuario Super Admin y verifica que llegas a <code>/admin</code> .
6	Cambiaste la contraseña del Super Admin	Nunca dejes <code>password</code> en producción.
7	La base usa un usuario dedicado	Revisa que <code>DB_USERNAME</code> no sea <code>root</code> .
8	Las copias automáticas funcionan	Ejecuta <code>/root/backup-suite.sh</code> a mano y revisa que se cree el archivo.
9	El cortafuegos está activo	<code>ufw status</code> — deben estar abiertos sólo los puertos 22, 80 y 443.
10	Creaste tu primera institución de prueba	Desde el panel Super Admin, y verificaste que su administrador puede ingresar.

6. Mantenimiento y actualizaciones en producción

6.1 Publicar una nueva versión del sistema

Cuando hagas mejoras en tu computadora y quieras llevarlas al servidor, este es el procedimiento seguro.

Rutina de actualización

```
# 1. Activa el modo mantenimiento (los usuarios ven un aviso)
$ php artisan down

# 2. Respalda la base ANTES de tocar nada
$ /root/backup-suite.sh

# 3. Trae los cambios del repositorio
$ git pull origin main

# 4. Actualiza las dependencias
$ composer install --optimize-autoloader --no-dev

# 5. Aplica los cambios de base de datos
$ php artisan migrate --force

# 6. Regenera las cachés
$ php artisan config:cache
$ php artisan route:cache
$ php artisan view:cache

# 7. Reactiva el sistema
$ php artisan up
```

6.2 Rutinas periódicas recomendadas

Frecuencia	Tarea
Diaria (automática)	Copia de seguridad de la base de datos mediante cron.
Semanal	Revisar <code>storage/logs/laravel.log</code> en busca de errores recurrentes.
Semanal	Descargar una copia del respaldo fuera del servidor.
Mensual	Actualizar el sistema operativo: <code>apt update && apt upgrade</code>
Mensual	Revisar el uso de disco: <code>df -h</code> y limpiar respaldos antiguos.
Trimestral	Verificar que la restauración funciona, probándola en un entorno de prueba.

6.3 Cuándo ampliar el servidor

Señal	Qué significa	Acción
El sistema tarda más de 3 segundos en cargar	El servidor está saturado.	Sube al siguiente plan de VPS.
<code>free -h</code> muestra memoria casi llena	Te faltan recursos de RAM.	Amplía el plan o revisa consultas ineficientes.
<code>df -h</code> supera el 80% de uso	El disco se está llenando.	Limpia respaldos antiguos o amplía el almacenamiento.
Más de 20 instituciones activas	Estás en el límite cómodo del KVM 2.	Considera migrar al KVM 4 u 8.

La buena noticia sobre escalar

Ampliar un VPS en Hostinger es un cambio de plan desde el panel: el servidor se reinicia con más recursos y no tienes que migrar nada. Tu sistema, tu dominio y tus datos siguen exactamente igual.

7. Anexo: comandos de referencia rápida

Comandos de Laravel (Artisan)

Comando	Qué hace
<code>php artisan serve --port=8062</code>	Levanta el servidor de desarrollo local.
<code>php artisan migrate</code>	Crea o actualiza las tablas de la base de datos.
<code>php artisan migrate --force</code>	Igual, pero sin pedir confirmación (para producción).
<code>php artisan migrate:fresh --seed</code>	Borra todo y reconstruye desde cero.
<code>php artisan db:seed</code>	Carga todos los datos de ejemplo.
<code>php artisan db:seed --class=X</code>	Carga sólo un seeder concreto.
<code>php artisan key:generate</code>	Genera la clave de cifrado de la aplicación.
<code>php artisan route:list</code>	Lista las 163 rutas con sus middleware.
<code>php artisan view:clear</code>	Limpia la caché de vistas compiladas.
<code>php artisan config:clear</code>	Limpia la caché de configuración.
<code>php artisan optimize</code>	Cachea configuración y rutas para producción.
<code>php artisan down / up</code>	Activa y desactiva el modo mantenimiento.

Comandos de Composer

Comando	Qué hace
<code>composer install</code>	Instala las dependencias según composer.lock.
<code>composer install --no-dev</code>	Igual, sin las herramientas de desarrollo (producción).
<code>composer update</code>	Actualiza las dependencias a versiones más nuevas.
<code>composer dump-autoload</code>	Regenera el cargador automático de clases.

Comandos del servidor Linux

Comando	Qué hace
<code>ssh root@IP</code>	Conectarse al servidor.
<code>systemctl reload nginx</code>	Recargar la configuración de Nginx.
<code>systemctl restart php8.2-fpm</code>	Reiniciar el procesador de PHP.
<code>nginx -t</code>	Verificar que la configuración de Nginx sea válida.
<code>df -h</code>	Ver el espacio en disco disponible.

Comando	Qué hace
<code>free -h</code>	Ver el uso de memoria.
<code>tail -f storage/logs/laravel.log</code>	Ver los errores de la aplicación en tiempo real.
<code>ufw status</code>	Ver el estado del cortafuegos.
<code>crontab -e</code>	Editar las tareas programadas.

Resumen en tres líneas

Desarrollo local: XAMPP + Composer + `php artisan serve`.

Producción: VPS Hostinger KVM 2 con Ubuntu, Nginx apuntando a `public/`, SSL de Let's Encrypt y copias automáticas por cron.

Regla de oro: `APP_DEBUG=false` y respaldo antes de cada cambio.

Sobre los precios de este documento

Las cifras de hosting corresponden a julio de 2026 y a contrataciones de largo plazo. Los proveedores cambian sus tarifas y promociones con frecuencia: **verifica siempre en el sitio oficial antes de contratar**. Los planes de renovación suelen costar más que el primer periodo.