

# Web Studio: integración de webs corporativas en una tienda online

## Informe técnico y funcional para evaluación editorial de Livecommerce.es

**Fecha de elaboración:** 20 de octubre de 2026.

**Estado del trabajo:** módulo con piloto local, administración y pruebas. Sin despliegue en producción acreditado.

**Finalidad:** conservar el conocimiento de la tarea y permitir a otra IA evaluar su interés divulgativo. Este documento no es un artículo de blog ni una certificación de seguridad o rendimiento.

## 1. Resumen de la tarea

Se ha desarrollado el módulo `web_studio` para incorporar páginas corporativas a una instalación de ShopinCloud/LiveCommerce. Permite subir varias webs, publicar versiones y seleccionar cuál sirve sus páginas junto a la tienda. Admite HTML convencional y un formato de plantillas Twig con cabecera, pie y otros componentes compartidos.

La necesidad funcional era gestionar contenidos como empresa, servicios o portada sin convertir cada página en una modificación del tema de ecommerce. El contenido pertenece al proyecto de la tienda, mientras que el módulo compartido contiene el código necesario para administrarlo y servirlo. Esta separación facilita trasladar los archivos del proyecto junto con sus webs.

El trabajo abarcó resolución de URLs, tratamiento de la portada, entrega de recursos, almacenamiento y publicación, subida desde el administrador, controles de acceso, diseño de las vistas y correcciones surgidas durante el piloto. Las últimas intervenciones mejoraron la información de los botones de estado y eliminaron la reordenación automática de la web activa.

El resultado es un piloto funcional. No se ha construido un editor visual de páginas ni se ha demostrado compatibilidad con cualquier web exportada por herramientas externas.

## 2. Contexto del ecommerce

- **Contenido corporativo y comercial:** ámbito principal. La tienda puede incorporar información sobre el negocio, sus servicios y su propuesta de valor dentro de la misma instalación.
- **SEO técnico y visibilidad:** se definieron URLs canónicas, redirecciones y convivencia con rutas existentes. No se midieron indexación, posiciones ni tráfico orgánico.
- **Experiencia de usuario:** se trabajó especialmente en el administrador, con estados comprensibles, enlaces de consulta, confirmaciones, errores visibles y posiciones estables de las tarjetas.
- **Seguridad y estabilidad:** se validan archivos y rutas, se restringen las operaciones de escritura y se protege el almacenamiento frente al acceso HTTP directo.
- **Eficiencia operativa y mantenimiento:** se incorporaron versiones, recuperación de la versión anterior, componentes Twig compartidos y almacenamiento vinculado al proyecto.

- **Escalabilidad:** se contemplan varias webs por proyecto y publicación con bloqueo. Esto no equivale a una validación de capacidad bajo alta concurrencia.
- **Rendimiento:** hubo integración con la caché para evitar contenido obsoleto. No fue una optimización de velocidad y no existen mediciones de mejora.

No se modificaron directamente catálogo, precios, stock, pedidos, pagos, facturación, ERP ni el proceso de compra. La relación con conversión es potencial e indirecta: disponer de mejores contenidos puede ayudar a explicar el negocio, pero esa consecuencia no se ha medido.

## 3. Problema original

### 3.1 Necesidad inicial

Se solicitó un módulo para alojar y seleccionar una web corporativa junto a la tienda, con rutas públicas comprensibles y sin duplicar cabecera y pie en cada plantilla. También se pidió poder cargar los archivos directamente, además de mediante ZIP, y utilizar el proyecto local real para las pruebas.

Había que resolver qué ocurriría si una página corporativa coincidía con una URL nativa del ecommerce. Se valoraron ambas prioridades y se acordó conservar la prioridad nativa, con una excepción explícita para la portada.

### 3.2 Incidencias y ajustes observados durante el piloto

- Las primeras rutas de ejemplo incluían un prefijo de piloto. Se pidió que la página de servicios se resolviera como `/servicios`, relativa a la raíz pública de la instalación.
- Se fijó que las páginas corporativas se sirvieran sin barra final y que sus variantes redirigieran a la URL canónica.
- Un ZIP aportado por el usuario produjo el error «El ZIP contiene rutas no permitidas». Posteriormente se ampliaron los nombres admitidos para incluir `+` y puntos internos, manteniendo los controles de rutas peligrosas.
- Se pidió admitir SVG, excluir XML y omitir los archivos inválidos sin abortar toda la importación por esa sola causa.
- El usuario informó de que un `index.html` situado en la raíz de la web no se servía como portada. El fallback del 404 no cubría por sí solo la ruta de inicio, ya resuelta por el controlador nativo.
- Se exigió utilizar vistas de administración `.tpl`, alojadas en `views` y cargadas mediante `$this->load->tpl()`, y compartir los estilos entre el listado y los formularios.
- Se comunicó que «Activar esta web» no funcionaba y no explicaba el motivo. En las comprobaciones posteriores se activaron correctamente las webs HTML y Twig, también después de navegación interna del administrador. No se reprodujo un fallo de persistencia de esa activación.
- El aviso de error estaba encima de las tarjetas y podía quedar fuera de la zona visible. Además, el controlador sustituía las excepciones por un mensaje genérico.
- Después, el usuario identificó que la web activada cambiaba de posición. El controlador efectivamente la colocaba en primer lugar. Se eliminó esa reordenación para conservar el orden de alta.

La secuencia respalda que el movimiento de las tarjetas generó confusión. No permite afirmar que todos los intentos anteriores fallaran por ese motivo ni atribuirlos a un error de servidor que no se consiguió reproducir.

### 3.3 Riesgos potenciales, no incidentes acreditados

Sin reglas de convivencia, una página corporativa podría ocupar una ruta importante de la tienda. Una publicación incompleta podría dejar páginas sin recursos; una caché desactualizada podría mostrar otra versión; y permisos incorrectos podrían impedir guardar cambios. También existen riesgos al ejecutar HTML o JavaScript de terceros en el mismo origen. No consta que esos riesgos causaran pérdidas comerciales, filtraciones o interrupciones en una tienda en producción.

## 4. Solución implementada

### 4.1 Rutas nativas y excepción de portada

La integración se realiza en `TPL_front_main`, dentro del núcleo `front` compartido. Al entrar en `show_404()`, se consulta el módulo antes de emitir el 404, aplicar sus redirecciones o registrar el error. El módulo comprueba si la solicitud corresponde a una página corporativa publicada y válida.

El controlador de Web Studio conserva el resultado del router: si ya existe una ruta nativa reconocida, una URL personalizada o un controlador nativo aplicable, no recupera esa solicitud como página corporativa. Los prefijos operativos reservados tampoco se aceptan como páginas corporativas. La comprobación de rutas personalizadas de base de datos se hace en ejecución; su detección completa al importar queda pendiente.

Para `/`, `TPL_front_main::home()` consulta `tryRenderHome()` antes de la redirección canónica de la tienda y del renderizado de CMS o tema. Si la web activa tiene una portada publicada y válida, se sirve. Si está deshabilitada, falta la portada o no supera la validación, continúa el comportamiento nativo. Esta es una excepción acordada a la prioridad nativa general.

### 4.2 Mapa de páginas y URLs canónicas

En HTML, el mapa se construye al publicar, a partir del inventario de archivos:

- `index.html` en la raíz corresponde a `/`.
- `servicios.html` corresponde a `/servicios`.
- `servicios/index.html` también corresponde a `/servicios`.
- `empresa/equipo.html` corresponde a `/empresa/equipo`.

Si `servicios.html` y `servicios/index.html` aparecen juntos, la publicación se rechaza porque ambos ocupan la misma URL. El sistema no elige arbitrariamente uno de ellos.

Las variantes con barra final, `.html` o `index.html` redirigen con HTTP 301 a la URL canónica y conservan los parámetros de consulta admitidos. La raíz `/` mantiene su barra. Una página inexistente continúa hacia el tratamiento nativo del 404.

Estas rutas son relativas a la raíz pública de la instalación. Un despliegue bajo un subdirectorio añade el `mount_path` configurado; desplegar en la raíz del dominio permite utilizar `/servicios` directamente. El prefijo de un entorno de desarrollo no debe confundirse con un requisito de la URL de negocio.

### 4.3 Almacenamiento, selección y versiones

La raíz lógica acordada es `frontR00TPATH . 'web_studio/'`. Dentro se conservan la configuración, el bloqueo de publicación, los temporales y una carpeta `sites/<identificador>/` por web. Cada web contiene borradores, temporales y versiones publicadas, con un inventario `release.json` y su contenido.

La persistencia del piloto utiliza `config.json`, sin tablas nuevas. Registra si el módulo está habilitado, la web activa, el prefijo público y las referencias de versiones. Se conserva una sola web activa por proyecto.

La publicación valida el paquete, prepara una instantánea completa y actualiza la configuración mediante un archivo temporal y `rename`, bajo bloqueo del proyecto. No sobrescribe una versión existente. Una nueva versión de una web inactiva no la activa; actualizar la activa cambia el contenido servido cuando finaliza la publicación. Se rechaza una subida que partía de una versión ya sustituida durante el envío.

Se pueden recuperar la versión anterior y desactivar el módulo conservando los archivos. Se comprueban tamaño y SHA-256 del contenido publicado: los cambios se realizan en borradores y se publican como otra versión. No se deben editar las versiones publicadas en caliente.

### 4.4 Entrega de archivos y adaptación de recursos

El almacenamiento está bloqueado para acceso HTTP directo mediante una protección local. Las páginas y los recursos se entregan desde el módulo; no se expone la carpeta de borradores ni la configuración como un directorio web público.

El gateway de recursos identifica web y versión y solo sirve archivos inventariados bajo `assets/`, con un tipo permitido. Utiliza URLs versionadas. Los recursos de versiones publicadas anteriores pueden seguir disponibles mientras el módulo esté habilitado, lo que ayuda a conservar referencias de páginas ya abiertas.

En HTML convencional, `DOMDocument` adapta las referencias inventariadas de `src`, `href` y `poster`, así como enlaces a páginas del paquete. Conserva las URLs externas y las rutas que no corresponden a archivos del inventario. SVG está admitido y se entrega como `image/svg+xml` con `nosniff`.

No está terminada la adaptación general de `srcset`, `url()` y `@import` de CSS, ni de rutas construidas desde JavaScript. Por ello, subir un paquete válido no demuestra que todos sus recursos funcionen sin ajustes.

### 4.5 Servicio Twig con componentes compartidos

El formato estructurado requiere `site.json`, con `format: shopincloud-site`, versión de formato 1 y un mapa de rutas a plantilla y título. Las páginas pueden usar `extends` e `include` para compartir layout, cabecera, pie y otros componentes. Una modificación del parcial se propaga a las páginas que lo incluyen al publicar una nueva versión.

Se reutiliza Twig 1.42.5 disponible en el repositorio, con carga desde un inventario en memoria, autoescape HTML, variables estrictas y una política sandbox limitada. No se ofrece un loader de archivos arbitrarios ni el contexto completo del ecommerce. El contexto corporativo incluye `page.title`, `page.url`, `site.name`, `site_base_url` y la función `asset()` para recursos inventariados.

Este renderizador es distinto del Twig de las vistas admin. En la administración se utiliza el loader nativo y se escapan expresamente los datos dinámicos con `|e`, porque su configuración general no

activa autoescape. La reutilización de Twig antiguo se limita al piloto y a paquetes de autores de confianza; no constituye una garantía para contenido no confiable.

## 4.6 Importación desde el administrador

Se admiten ZIP, selección de una carpeta completa y varios archivos. La carpeta conserva sus subdirectorios; la selección de archivos sueltos los coloca en la raíz. Un ZIP puede contener una única carpeta envolvente. Subir una nueva versión requiere la web completa: no mezcla archivos con la versión anterior.

La carga secuencial incluye progreso, cancelación y validación. Los límites del piloto son 500 archivos, 10 MiB por archivo y 50 MiB totales, sujetos también a los límites efectivos de PHP. ZIP requiere ZipArchive.

Los nombres admiten letras, números, guion, guion bajo, punto y +, manteniendo el rechazo de rutas absolutas, segmentos de escape, caracteres de control, archivos ocultos y enlaces simbólicos. En la subida admin se omiten los archivos no admitidos, incluidos PHP, ejecutables y XML; también se omiten entradas inválidas del ZIP y metadatos del sistema operativo. Los archivos válidos pueden continuar.

La pantalla informa de archivos guardados y omitidos, con nombre y motivo de hasta los primeros 100 descartes. Si no queda ningún archivo válido, termina sin publicar. Un ZIP ilegible, una web sin entrada válida, conflictos de rutas o un fallo de almacenamiento siguen impidiendo publicar. La validación estricta de carpetas publicadas por CLI se mantiene.

## 4.7 Administración, autorización y respuesta de los controles

El listado y los formularios utilizan .tpl dentro de admin/views, cargadas con `$this->load->tpl()`, y CSS compartido. Se respetan los patrones del módulo y referencias de administración como `file_explorer`, `customer_ticket`, `products_complementary` y `themes`.

Las tarjetas muestran formato, versión, número de archivos y páginas. Solo las rutas de la web activa enlazan al front; las demás quedan como información. Las miniaturas son ilustrativas y no capturas automáticas de cada web.

Activar una web mantiene ahora el orden de alta. La tarjeta cambia de estado y resaltado sin desplazarse al primer puesto. Mientras se guarda, el botón muestra progreso y se bloquean otras operaciones de estado de ese listado. El error aparece junto al botón, recibe el foco y se lleva a la zona visible; después se puede reintentar. Al guardar correctamente, se muestra una confirmación en el listado.

El controlador devuelve motivos de validación y mensajes comprensibles para fallos de almacenamiento, sin exponer las rutas absolutas de los avisos PHP. Antes de operar comprueba sesión, usuario, proyecto cargado, permisos del módulo y CSRF. Los lotes de subida están vinculados a sesión, formulario y web. Las respuestas JSON fuerzan la persistencia de las escrituras de sesión diferidas antes de terminar.

## 4.8 Convivencia con la caché

Se añadió una comprobación en `ED_Output::_display_cache()`, también utilizada en el arranque temprano del front. Evita leer la caché completa nativa para solicitudes candidatas de Web Studio habilitado. La respuesta corporativa termina sin envolverla en el layout nativo ni escribirla en esa caché.

El piloto desactiva la caché de compilación Twig y solicita que las respuestas no se almacenen. La documentación del módulo registra una particularidad de Apache local: sustituye Cache-Control, por lo que también se envía una fecha Expires vencida. Las cabeceras efectivas deben verificarse en cada despliegue. Esta decisión favorece la consistencia de versiones durante el piloto; no acredita menor tiempo de respuesta.

## 5. Impacto para una tienda online

**Capacidad funcional verificada:** gestionar varias webs corporativas, publicar sus versiones y seleccionar una dentro del administrador. El responsable dispone de enlaces para consultar las páginas de la activa y de información sobre el resultado de las operaciones.

**Eficiencia operativa esperable:** reutilizar cabecera y pie puede reducir cambios repetidos. La carga de carpetas evita preparar un ZIP cuando no sea necesario. Omitir archivos irrelevantes permite continuar con los válidos y proporciona una explicación del descarte. No se han medido horas ahorradas ni costes de mantenimiento.

**Reducción de errores esperable:** separar subida y activación, disponer de una versión anterior y conservar la posición de las tarjetas ayuda a entender qué contenido está publicado. Las confirmaciones y errores próximos al control reducen la ambigüedad. No existe un estudio de usuarios que cuantifique esa mejora.

**Compra y conversión:** contenidos corporativos coherentes podrían reforzar la comprensión del negocio y la confianza. No se realizaron pruebas A/B ni se analizaron ventas, abandono o solicitudes comerciales. No se puede atribuir una mejora de conversión al módulo.

**Fiabilidad y crecimiento:** validaciones, bloqueo y cambio de referencia de versión aportan mecanismos de control. Antes de ampliar su uso hacen falta políticas de conservación, revisión de seguridad y pruebas de concurrencia y capacidad. El almacenamiento por proyecto favorece la organización y portabilidad, pero no sustituye copias de seguridad ni constituye una migración automática.

## 6. Resultados y verificaciones

### 6.1 Hechos verificados en esta tarea

- tests/run.php: **93 aserciones correctas** en la ejecución realizada al elaborar este informe. Ejecuta Twig real, la integración con front\_main y el tratamiento de caché temprana, sin arrancar la aplicación completa ni abrir la base de datos.
- tests/uploads.php: **112 comprobaciones correctas** durante la corrección de los controles de estado. Cubre importación ZIP y directa, límites, omisión de archivos, propiedad del lote, conservación de borradores, publicación, recuperación y renderizado de vistas admin.
- tests/admin\_access.php: **21 comprobaciones correctas** durante esa misma corrección. Incluye identidad, permisos, CSRF, sesión diferida, fronteras de subida HTTP, activación, mensajes de error y conservación de configuración ante fallos.
- En total, esos resultados suman **226 aserciones o comprobaciones de las tres suites**, ejecutadas en los momentos indicados. No representan 226 pruebas completas de navegador ni una auditoría de seguridad.

- Se comprobaron sintaxis PHP y JavaScript tras los cambios de feedback. La eliminación posterior de la reordenación pasó el lint PHP y se verificó visualmente.
- El catálogo documental pasó su comprobación de firmas, referencias y enlaces locales durante la tarea.
- En el administrador local se activaron webs HTML y Twig, se observó la confirmación y se comprobó el acceso a rutas de la web activa. Se probó también después de navegación interna del admin.
- Una simulación de fallo de conexión en la pestaña de prueba mostró el error junto al botón, con foco y posibilidad de reintento. Se retiró el bloqueo temporal de red y el reintento activó la web correctamente.
- Tras eliminar la reordenación, la revisión del listado mostró la web activa en la segunda posición, conservando el orden de alta.

Las suites PHP que publican, alteran archivos o cambian de web operan sobre copias temporales; los borradores del proyecto real se utilizan como fuente de lectura. No se ejecutaron esas alteraciones de prueba sobre el contenido publicado real.

## 6.2 Evidencia adicional y límites de su interpretación

El README registra comprobaciones HTTP anteriores contra Apache local para páginas, recursos, redirecciones, portada, 404 y bloqueo de almacenamiento privado. Incluye un script de comprobación de solo lectura. No se ha vuelto a ejecutar esa batería HTTP completa para redactar este informe; esa evidencia debe distinguirse de los resultados de ejecución enumerados arriba.

No se consiguió reproducir el fallo original de activación comunicado por el usuario. Sí se verificaron y corrigieron problemas de información y estabilidad visual. La última observación del usuario explica la confusión por reordenación, pero no aporta una traza de error de servidor.

## 6.3 Beneficios esperables y aspectos no comprobados

Son razonablemente esperables una administración más comprensible, menor repetición al mantener componentes y mayor control del cambio de versiones. Están pendientes mediciones de usabilidad, tiempo de gestión, rendimiento, comportamiento bajo carga, compatibilidad general con paquetes externos y resultados SEO o comerciales. Tampoco se ha verificado el funcionamiento en una infraestructura de producción.

# 7. Aprendizajes transferibles

1. **Definir quién resuelve cada URL antes de integrar contenido nuevo.** Una tienda combina catálogo, cuenta, compra y contenidos. La prioridad y las excepciones deben ser explícitas, especialmente en la portada.
2. **Distinguir archivos descartables de fallos que impiden publicar.** Un archivo auxiliar no permitido no tiene por qué inutilizar los archivos válidos, pero una web incompleta o con rutas ambiguas no debería publicarse silenciosamente.
3. **Separar preparación, publicación y activación.** Ayuda a revisar alternativas sin convertir cada subida en un cambio visible de la tienda. Conservar versiones aporta capacidad de recuperación, aunque sigue siendo necesaria una política de copias y retención.

4. **Tratar la estabilidad visual como parte del funcionamiento.** Un cambio de posición puede hacer que una acción correcta parezca fallida. La señal útil es que el usuario repite clics o busca el resultado en el lugar anterior.
5. **Mostrar la respuesta cerca de la acción.** Un error fuera de la zona visible equivale, para el usuario, a una falta de respuesta. El progreso, la confirmación y la recuperación tras fallo deben formar parte del diseño del control.
6. **Comprobar el paquete servido, no solo el paquete admitido.** Una importación puede superar la validación y conservar referencias de recursos que requieren adaptación. Revisar páginas y recursos reales es una fase distinta de validar extensiones.
7. **Alinear versiones y caché.** Si el administrador confirma un cambio y el visitante sigue viendo contenido anterior, deben revisarse tanto el estado publicado como las capas de caché y las cabeceras efectivas.

Estos patrones pueden aparecer en otras tiendas con páginas corporativas, campañas, módulos de contenidos o gestores de varias versiones. No dependen exclusivamente de ShopinCloud.

## 8. Potencial divulgativo

**Enfoque 1: convivencia entre tienda y web corporativa.** Explicar a un responsable de ecommerce cómo incorporar empresa o servicios conservando las rutas de compra y catálogo. Material disponible: prioridad nativa, excepción de portada y ejemplos de URLs. Límite: no presentarlo como una mejora SEO medida ni como compatibilidad universal.

**Enfoque 2: cambios de contenido con control de versiones.** Mostrar la diferencia entre subir una web, publicarla y activarla, junto con la posibilidad de recuperar la versión anterior. Material disponible: flujo del admin, validaciones y omisiones explicadas. Límite: no equiparar el rollback a una copia de seguridad ni prometer ausencia total de interrupciones.

**Enfoque 3: cuando una interfaz hace parecer que una acción ha fallado.** Analizar el efecto de mover la tarjeta activa y ocultar el feedback fuera de la zona visible. Material disponible: incidencia comunicada, activaciones comprobadas, observación posterior del usuario y corrección del orden. Límite: no afirmar que se descubrió una avería de servidor ni cuantificar reducción de errores sin estudio.

**Valoración editorial:** existe material útil para explicar decisiones operativas reales del ecommerce. El caso tiene más fuerza como explicación de diseño, publicación y gestión de contenidos que como relato de aumento de ventas o rendimiento. Una publicación debe describir el alcance de piloto.

## 9. Limitaciones y confidencialidad

### 9.1 Alcance pendiente o no acreditado

- No hay despliegue en producción acreditado, métricas comerciales, medición de rendimiento ni evaluación de carga.
- Quedan pendientes previsualización autenticada de webs inactivas y edición de archivos desde el navegador.
- No existe limpieza automática de temporales abandonados ni de versiones antiguas. Se necesita definir retención, capacidad y mantenimiento.

- No se ha completado la detección de conflictos con todas las rutas de base de datos durante la importación.
- La adaptación de recursos de paquetes arbitrarios tiene los límites descritos en la sección 4.4.
- El piloto utiliza Twig antiguo y admite HTML, JavaScript y SVG de autores de confianza. Sus controles no acreditan aislamiento seguro de autores no confiables.
- El bloqueo de almacenamiento y los permisos de lectura/escritura deben validarse en cada servidor. Los mecanismos locales no demuestran protección equivalente en cualquier infraestructura.
- XML, incluido `sitemap.xml`, se excluye de estas subidas por decisión funcional. No se ha implementado dentro de este trabajo una estrategia completa de sitemaps, canonicals HTML, datos estructurados o migración SEO.
- No se conocen el sector comercial, volumen de catálogo, tráfico, presupuesto, organización ni necesidades de producción de una tienda destinataria.

## 9.2 Datos que deben excluirse de una publicación

Este informe omite credenciales, tokens, identificadores de sesión, nombres de personas, nombre del proyecto piloto, rutas absolutas de máquinas y datos de clientes o ventas. Una publicación pública tampoco debe incorporar volcados de configuración, profiler, logs, permisos efectivos del servidor, capturas sin anonimizar ni detalles de topología interna.

Los nombres de clases y las referencias de código siguientes sirven para trazabilidad técnica de este informe. En una pieza divulgativa conviene sustituirlos por explicaciones funcionales, salvo que su mención sea necesaria y esté autorizada. No se incluyen capturas del admin para evitar reutilizar accidentalmente información identificativa del entorno de prueba.

## 9.3 Fuentes de trabajo y mapa técnico

La base del informe es la conversación de desarrollo, el código y la documentación del módulo revisados al elaborarlo, las salidas de pruebas disponibles y las observaciones del navegador registradas durante la tarea. No se han incorporado afirmaciones comerciales externas ni investigación de mercado.

Referencias del repositorio, expresadas de forma relativa y sin datos de instalación:

- `modules/web_studio/README.md`: alcance, decisiones, uso, restricciones y comprobaciones documentadas.
- `modules/web_studio/libraries/WebStudioSite.php`: resolución, normalización, integridad y entrega de páginas y recursos.
- `modules/web_studio/libraries/WebStudioPublisher.php`: publicación, bloqueo, selección y recuperación de versiones.
- `modules/web_studio/libraries/WebStudioUpload.php`: importación y validación de archivos y lotes.
- `modules/web_studio/libraries/WebStudioTwig.php`: renderizado corporativo con contexto y política limitados.
- `modules/web_studio/front/controllers/web_studio.php`: integración de portada y fallback con la petición front.

- `front_core/core_v1/modules/front_main/front/controllers/TPL_front_main.php`: puntos de entrada `home()` y `show_404()`.
- `application/core/ED_Output.php`: tratamiento de la caché completa antes de servir contenido.
- `modules/web_studio/admin/controllers/web_studio.php`: permisos, formularios, operaciones de estado y respuestas.
- `modules/web_studio/admin/views/index.tpl`, `upload.tpl` y `_icons.tpl`: vistas del listado y de subida.
- `modules/web_studio/admin/assets/web_studio.css` y `web_studio.js`: presentación compartida, envío y feedback.
- `modules/web_studio/tests/`: suites aisladas de front, subidas y acceso admin, más comprobaciones HTTP.

Las referencias describen el estado del trabajo revisado. No prueban por sí mismas un despliegue ni sustituyen las verificaciones del entorno donde se vaya a utilizar el módulo.