EcomLabs

GUÍA PARA DECIDIR

PWA o app nativa para tu tienda

Una tabla para decidir si tu tienda necesita una app nativa o le basta una PWA, y la checklist mínima de la PWA: manifest, service worker, HTTPS, iconos, página sin conexión y medición en GA4. Con el código que probamos.

PARA	Tiendas en línea que evalúan tener una app
PROBADO CON	Chromium 151 (Playwright 1.62) · Node 22.22.3 · Windows 11
ACTUALIZADA	3 de noviembre de 2026
DOCUMENTACIÓN	web.dev/learn/pwa
ARTÍCULO	ecomlabs.dev/g/pwa
AUTOR	Luis Morales · @luismoralesdev

Índice

Antes de empezar	Qué decide esta guía
Paso 01	Decide: PWA o app nativa
Paso 02	El manifest
Paso 03	El service worker y su registro
Paso 04	HTTPS, iconos y página sin conexión
Paso 05	Mide instalaciones y aperturas
Al terminar	Checklist mínima de la PWA

ANTES DE EMPEZAR

Qué decide esta guía

Si tu tienda necesita una app nativa o le basta una PWA, y qué tiene que tener la PWA mínima. Una PWA es tu misma tienda con tres piezas más: HTTPS, un manifest y un service worker. Se instala desde el navegador, responde sin conexión y, en iPhone, recibe push si el cliente la agrega a la pantalla de inicio.

El código de esta guía lo probamos el 2 de noviembre de 2026 con Chromium 151 (Playwright 1.62) y Node 22.22.3 en Windows 11: el service worker controla la página tras recargar, Chromium lee el manifest sin errores ni errores de instalabilidad, y sin red una página no visitada devuelve `offline.html`. No probamos una instalación real ni un iPhone.

PASO 01

Decide: PWA o app nativa

Responde cada fila. Si casi todo cae en la columna de la PWA, empieza por ella y mide. Es nuestra regla práctica, no una regla de Apple ni de Google.

CRITERIO	PWA BASTA SI	APP NATIVA SI
Cómo llegan	Google, anuncios, redes	Abren la app desde el icono
Recompra	Cada varios meses	Cada semana o cada mes, con cuenta
Push en iPhone	Un extra (solo web apps de inicio, iOS 16.4+)	Parte central del negocio
Funciones	Catálogo, carrito, checkout	Live Activities (ActivityKit) u otras solo nativas
Tienda de apps	No es tu canal (una PWA también puede listarse)	Tu público busca en App Store o Google Play
Equipo	El que ya mantiene la web	iOS, Android y revisiones de cada tienda
Costo fijo	Tu hosting de siempre	Apple: 99 USD por año de membresía · Google Play: 25 USD una vez

PASO 02

El manifest

Guárdalo como `manifest.webmanifest` en la raíz, sírvelo como `application/manifest+json` y enlázalo en cada página con `<link rel="manifest">`:

```
{
  "id": "/",
  "name": "Tienda Demo",
  "short_name": "Demo",
  "start_url": "/?source=pwa",
```

```
  "scope": "/",
  "display": "standalone",
  "background_color": "#ffffff",
  "theme_color": "#151412",
  "icons": [
    { "src": "/icons/icon-192.png", "sizes": "192x192", "type": "image/png" },
    { "src": "/icons/icon-512.png", "sizes": "512x512", "type": "image/png" }
  ]
}
```

- **Lo que pide Chrome** (web.dev): `short_name` o `name`, iconos de 192 y 512 px, `start_url`, `display` en `fullscreen`, `standalone`, `minimal-ui` o `window-controls-overlay`, y `prefer_related_applications` ausente o en `false`.
- `id`: si falta, la identidad es `start_url` (MDN) y cambiarlo crea otra app. Si ya tienes una PWA instalada sin `id`, fija `id` con tu `start_url` actual antes de agregar `?source=pwa`.
- `?source=pwa` en `start_url` marca las aperturas desde el icono para medirlas.

PASO 03

El service worker y su registro

Guárdalo como `/sw.js`, en la raíz del dominio: su alcance por defecto es la carpeta donde vive. Guarda en caché solo el esqueleto; las páginas van a la red y, sin red, responde la página sin conexión:

```
const CACHE = "shell-v1";
const SHELL = ["/offline.html", "/app.css", "/icons/icon-192.png"];

self.addEventListener("install", (event) => {
  event.waitUntil(caches.open(CACHE).then((cache) => cache.addAll(SHELL)));
});

self.addEventListener("activate", (event) => {
  event.waitUntil(
    caches.keys().then((keys) => Promise.all(keys.filter((k) => k !== CACHE).map((k) => caches.delete(k)))),
  );
});

self.addEventListener("fetch", (event) => {
  const { request } = event;
  const url = new URL(request.url);
  if (request.mode === "navigate") {
    event.respondWith(fetch(request).catch(() => caches.match("/offline.html")));
  } else if (request.method === "GET" && url.origin === location.origin && SHELL.includes(url.pathname)) {
    event.respondWith(caches.match(request).then((hit) => hit || fetch(request)));
  }
});
```

Regístralo al inicio de `pwa.js`, cargado en todas las páginas. La primera visita lo instala; la página queda controlada desde la siguiente carga:

```
if ("serviceWorker" in navigator) {
  navigator.serviceWorker.register("/sw.js");
}
```

> **CUIDADO**
>
> No guardes en caché el HTML de productos, el carrito ni el checkout: un cliente podría ver un precio viejo o un agotado como disponible. Y cuando cambies un archivo del esqueleto, sube el nombre de la caché (`shell-v2`) para que `sw.js` cambie y el navegador instale la versión nueva.

PASO 04

HTTPS, iconos y página sin conexión

- **HTTPS** en toda la tienda. Para desarrollar, `localhost` y `127.0.0.1` cuentan como contexto seguro.
- **Iconos** PNG de 192×192 y 512×512 en las rutas del manifest.
- `offline.html`: una página corta, sin precios ni stock, que use solo archivos del esqueleto:

```
<!doctype html>
<html lang="es">
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Sin conexión</title>
<link rel="stylesheet" href="/app.css">
</head>
<body><h1>Sin conexión</h1><p>Revisa tu conexión y vuelve a intentarlo.</p></body>
</html>
```

PASO 05

Mide instalaciones y aperturas

El resto de `pwa.js`. Usa el `gtag` de tu etiqueta de GA4:

```
window.addEventListener("appinstalled", () => {
  gtag("event", "pwa_install");
});

const standalone = window.matchMedia("(display-mode: standalone)").matches;
const fromIcon = new URLSearchParams(location.search).get("source") === "pwa";
if (standalone && fromIcon) {
  gtag("event", "pwa_launch");
}
```

- `pwa_install` solo llega desde navegadores Chromium: `appinstalled` no existe en Safari.

- `pwa_launch` cuenta aperturas desde el icono en cualquier plataforma donde la página se abra con `display-mode: standalone`. No lo probamos en un iPhone: compruébalo en DebugView.
- En GA4, crea en Explorar un segmento de sesiones que incluyan `pwa_launch` y compáralo con el resto: compras, valor y cada cuánto vuelven.

AL TERMINAR

Checklist mínima de la PWA

- [ ] Toda la tienda se sirve con HTTPS.
- [ ] El manifest tiene `id`, `name`, `short_name`, `start_url` con `?source=pwa`, `scope` y `display: "standalone"`.
- [ ] Los iconos de 192 y 512 px existen en las rutas del manifest.
- [ ] Cada página enlaza el manifest y el servidor lo entrega como `application/manifest+json`.
- [ ] `/sw.js` vive en la raíz y se registra en todas las páginas.
- [ ] La caché guarda solo el esqueleto y `offline.html`; HTML, carrito y checkout van a la red.
- [ ] Sin red, una página no visitada muestra `offline.html`.
- [ ] El nombre de la caché cambia con cada cambio del esqueleto.
- [ ] GA4 recibe `pwa_install` y `pwa_launch`, comprobado en DebugView.
- [ ] La decisión de la app nativa se toma con el segmento de `pwa_launch`, no antes.