

Lovable + WooCommerce: implementation guide

A custom website, connected to your commerce platform

A Lovable React/TanStack website is not a Liquid theme or a WordPress theme. Keep it hosted as a separate frontend and connect the commerce platform through its supported APIs. This guide describes the architecture; a developer must implement and test the connections for your store.

1. Choose the architecture

Keep WordPress and WooCommerce running on a compatible host. Enable working REST endpoints and non-plain permalinks. Audit your payment, shipping and checkout plugins: not all support a separate frontend. A native WordPress theme is a different project, not an automatic conversion of your Lovable site.

2. Connect products safely

Use the public WooCommerce Store API at `/wp-json/wc/store/v1` for customer-facing products, cart and checkout. Privileged store management uses `/wp-json/wc/v3` through authenticated server-side code. Never expose WooCommerce consumer secrets in the browser. Configure cross-origin access only for your frontend when needed.

3. Implement the cart and checkout

Read the Store API cart and use its Cart-Token for the same buyer's cart operations, or a valid nonce where appropriate. Submit checkout through the Store API only with supported payment-gateway integration and required `payment_data`. Redirecting to the existing WordPress checkout does not automatically transfer a headless cart: implement and test a session/cart handoff, or keep the native store journey.

4. Host and launch

Deploy the Lovable frontend to a host compatible with its actual runtime. Keep the commerce platform online separately. Use HTTPS, configure domains and server-side secrets, and verify product links, images, canonical URLs, redirects and language pages. Preserve existing URLs or add 301 redirects before replacing a storefront.

5. Test before going live

Place a test order from product selection to payment confirmation. Test stock changes, variants, tax, shipping, currency, discounts, failed payment and order emails. Check mobile layouts, consent and analytics. Test each supported language. Confirm your team can update products without editing the frontend.

Costs and compatibility

Shopify subscriptions, payment fees, hosting and any paid plugins are separate costs. WooCommerce extensions may depend on WordPress-rendered checkout or sessions. A headless website does not automatically inherit every app, plugin, theme feature or customer account flow. Confirm compatibility before committing to the build.

Official implementation references

<https://developer.woocommerce.com/docs/apis/store-api/>

<https://developer.woocommerce.com/docs/apis/store-api/cart-tokens/>

<https://developer.woocommerce.com/docs/apis/store-api/checkout/>

<https://woocommerce.github.io/woocommerce-rest-api-docs/>

Need help planning the connection? support@asia2eu.eu

Reviewed 12 October 2026 · General guidance, not a turnkey plugin