

Lovable + Shopify: 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

Use an existing Shopify store. Keep product management, inventory and orders in Shopify. First assess whether your apps, subscriptions, shipping and account features support a headless storefront. Alternatively, keep the existing store and use Lovable for separate marketing pages.

2. Connect products safely

In Shopify admin, configure the Headless sales channel and the required Storefront API permissions. Connect using the Storefront GraphQL API and a supported API version. A public Storefront token may be used in the browser where appropriate; private Storefront tokens and Admin API credentials must remain server-side. Request only necessary access.

3. Implement the cart and checkout

Implement `cartCreate`, `cartLinesAdd` and `cartLinesUpdate`, plus quantity, variant and removal states. Read the `cart checkoutUrl` and send the buyer to Shopify's hosted checkout. Do not build your own payment form using an Admin token. Test discount codes, currencies, shipping and customer accounts with your store configuration.

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://shopify.dev/docs/storefronts/headless>

<https://shopify.dev/docs/api/storefront/latest/mutations/cartCreate>

<https://shopify.dev/docs/storefronts/headless/building-with-the-storefront-api/cart/manage>

Need help planning the connection? support@asia2eu.eu

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