Your Expo app is built. Get it ready for App Review.

The Expo App Store Submission Playbook

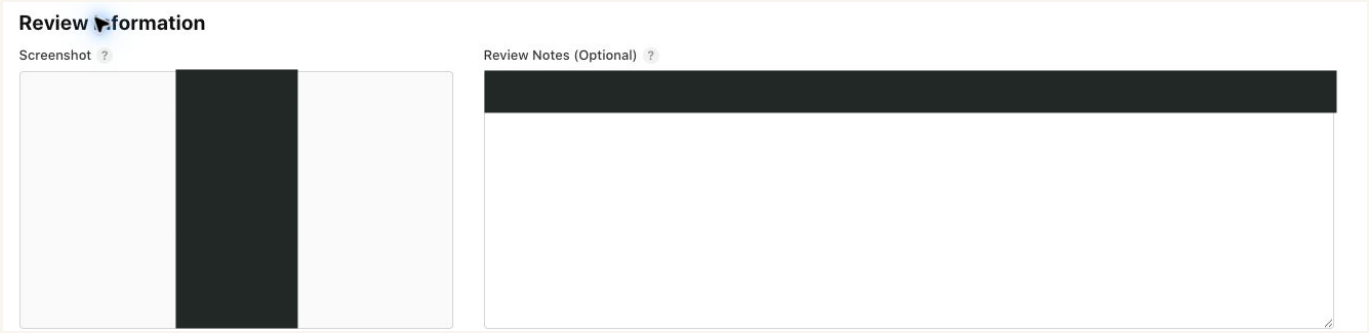
PDF playbook + agent audit pack for the last mile.

VERTO / EDITION 1.1 / SEPTEMBER 2026

42 PAGES + AGENT PACK

Include the first purchase in the submission.

Creating a product and submitting it are separate actions.



Purchase review information: the screenshot should show the real paywall; notes should give the exact route to it. Fictional route: Settings > Upgrade > Monthly. This record is already approved.

The common trap

The app's paywall works in development. Products exist in App Store Connect. The first submission still does not include those products for Apple to review.

1. Open the app version

Open the version you intend to send. Find the In-App Purchases and Subscriptions section. Use the available add/select control to include the first products with that version.

2. Resolve missing information

If a product is absent from the picker, inspect its status and required fields. For subscriptions check group localization as well as the product's own review information and screenshot.

3. Verify the actual selection

Return to the app version. Confirm the intended product IDs are listed before submitting the version. Do not infer this from a provider dashboard or source configuration.

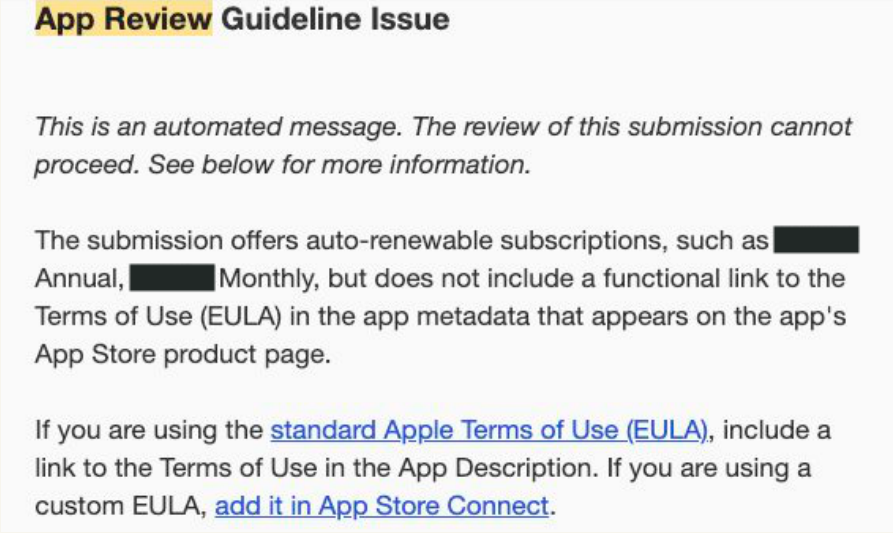
Done looks like

Your first in-app purchase is part of the version's submission. Later products may be submitted separately after the first has been approved; follow the workflow Apple shows for their current status.

Reference: https://developer.apple.com/help/app-store-connect/manage-submissions-to-app-review/submit-an-in-app-purchase/

P8 / Terms missing from store metadata

The app has links but the submission metadata is incomplete.



App Review Guideline Issue

This is an automated message. The review of this submission cannot proceed. See below for more information.

The submission offers auto-renewable subscriptions, such as [redacted] Annual, [redacted] Monthly, but does not include a functional link to the Terms of Use (EULA) in the app metadata that appears on the app's App Store product page.

If you are using the standard Apple Terms of Use (EULA), include a link to the Terms of Use in the App Description. If you are using a custom EULA, add it in App Store Connect.

Apple email / app identity withheld / September 2026 · Original email crop; identifying details redacted.

Evidence / Apple email / app identity withheld / September 2026

Automated review message: Terms of Use missing from product-page metadata.

What this proves - and what it does not

The email distinguishes the standard Apple EULA from a custom EULA. It is not asking for a differently styled paywall or another price test. The useful next step is to inspect the product-page metadata and the selected license arrangement, then open the resulting link without a private session.

Investigate

For a standard EULA, follow Apple's direction to include the terms link in the description. For a custom license, inspect the relevant App Store Connect field. Check the correct locale and saved version. Do not claim that adding a footer elsewhere on the web satisfies a missing App Store field.

Fix and close the loop

Illustrative response: "The [locale] description now includes [terms URL], which opens publicly. We re-opened the saved field and checked the product-page metadata. [Build change, if any]." The screenshot shows the concern; it does not establish that this response was sent or accepted.

Reference: https://developer.apple.com/app-store/review/guidelines/

Read a finding like an operator.

Three real rows from an earlier Claude Code run; explanations below are editorial.

IDENTITY-001 / VERIFIED_PASS / blocker

"name, slug, version, ios.bundleIdentifier and ios.buildNumber all present and non-empty." Evidence: app.json. This proves that the named fields existed in that snapshot. It does not prove that the uploaded binary or store record matches them. Compare those external identities using the release record.

COMPAT-002 / VERIFIED_FAIL / warning

"patchedDependencies pins react-native@0.79.2, but 0.81.5 is installed; one dependency is a different SDK major from the rest." Evidence: package.json. Inspect whether the stale patch is actually applied and whether the dependency set is coherent before changing versions. Do not perform a broad upgrade solely to make a warning disappear.

ARTIFACT-001 / UNRESOLVED / blocker

"No .ipa or .app supplied, so DTPlatformName cannot be verified. Provide the exact submission artifact." The next action is to obtain evidence from the intended build, not ask the agent to guess from a simulator run. Preserve the original unresolved result until a relevant artifact is inspected.

Provenance and current scope

Transcribed from the archived v0.1.1 run on a real Expo repository, with app identity withheld. That run covered 28 manifest checks: 11 passed, 2 failed and 15 remained unresolved. The included v0.3.2 pack contains 31 checks. These rows illustrate the report format; they are not a new validation run of v0.3.2.