

Read this before you hit Submit for Review.

Twelve things to check the night before your first App Store submission. Three of them are mine, learned by arguing with Apple. The other nine are the ones that catch everybody else.

BY DAVE BROWN

Nobody gets rejected for the thing they were worried about. You spend six months on the hard part, the sync engine, the offline mode, the thing that actually took skill, and the rejection comes back about a button that was four pixels under the home indicator on an iPad nobody on your team owns.

I've been through this. Two of the items below cost me a round trip with App Review each, and in both cases I'd have called the problem obvious if it had been someone else's app. That's the part worth internalising: these aren't hard. They're just invisible from the inside, because you know what your app is for and the reviewer doesn't.

THE MENTAL MODEL EVERYTHING ELSE FOLLOWS FROM

One person will open your app. They have a queue behind them. They did not read your landing page, your pitch deck, or your onboarding copy, and they will not give you the benefit of the doubt on anything ambiguous, because giving people the benefit of the doubt is how bad apps get through and that's the mistake that costs them, not yours.

So the question to ask about every screen isn't *is this fair* or *is this technically compliant*. It's *could a tired stranger with no context misread this in a way that ends badly for me*. Nuance is a luxury of people who already understand the product.

One

Reviewer, one pass, working from whatever is on screen and in your review notes.

90%

Of submissions reviewed within 24 hours, per Apple. Rejections are cheap in money, expensive in days.

5.1.1(v)

The guideline number for account deletion. Memorise it; it's the one that got me.

iPad

The device they'll test on that you almost certainly haven't.

Two housekeeping notes. First, I've put the guideline number on each item, because a rejection arrives as a number and a paragraph of boilerplate, and knowing which number maps to which real-world mistake saves you an afternoon of squinting. Second, account deletion appears once, at the top, rather than twice. It's both my hardest-won lesson and one of the most common rejections going, and repeating it wouldn't make it more true.

— WHAT'S IN HERE

- 01 **The three that got me.** Deleting data, the iPad, and explaining permissions like the reader is five.
 - 02 **The nine they reject everybody for.** Completeness, crashes, privacy policy, metadata, paywalls, UI, login walls, permission priming, and originality.
 - 03 **Submit day.** A review-notes template you can paste, and a one-page table to run down before you click the button.
-

Nothing clever here. Just the three I'd check first, in order.

Two of these I went back and forth with Apple on personally. The third is the one I now check before anything else, because it takes ten minutes and it is the single most common way a perfectly good app looks broken to a stranger.

01

A delete button a tired stranger can find in ten seconds.

GUIDELINE 5.1.1(V) · ACCOUNT DELETION

If your app lets someone create an account, it has to let them delete it from inside the app, without emailing you, without a web form, without a support ticket, and without a phone call. That much is widely known. What actually gets people rejected is the distance. The flow exists, but it's behind Settings, then Privacy, then Advanced, then a screen called *Manage your information*.

Put it where a reviewer looking for it will look: Profile or Settings, top level, one screen deep, and call it **Delete Account**. Not "Manage data." Not "Privacy options." The label is the whole test. If someone has to read three words of explanatory copy to know that's the button, you've already failed the ten-second version of this.

On your side, do a soft delete. Set a `deleted_at`, cut the account out of every read path immediately, and keep the rows for a disclosed window, usually thirty days, so you can recover an account for someone who fat-fingered it at 1am, and so your fraud, billing and accounting history doesn't develop holes. That is compatible with the guideline. What isn't compatible is a soft delete that never becomes a hard one.

THE DISTINCTION PEOPLE GET WRONG

Deactivation is not deletion. If your button sets `active = false` and that's the end of the story, you are out of compliance no matter what the button says, and if a reviewer or a user ever pokes at it you'll deserve what happens. The rule I'd hold yourself to: the screen tells the user plainly what goes away and when, the grace window is stated, and something in your system actually purges at the end of it. Write the purge job before you ship the button, not after.

One more that catches people: if you offer **Sign in with Apple**, deleting an account also means revoking that user's token through Apple's revocation endpoint. Apple requires it, it's a server-side call most teams have never made, and it isn't optional just because your app otherwise behaves.

WHERE IT LIVES

Profile or Settings, top level.
One tap from a screen they'll definitely open.

THE TEST

Hand your phone to someone who's never seen the app: "delete your account."
Time them.

ALSO REQUIRED

Full account deletion, not just in-app data. Plus SIWA token revocation if you use it.

02

Open your app on an iPad before Apple does.

GUIDELINE 2.1 · APP COMPLETENESS · AND 4.0 DESIGN

You built on an iPhone simulator, you tested on your own phone, and your whole team has the same three devices. Review does not. If your app is marked as supporting iPad, they will run it on an iPad, and if it isn't, there's still a good chance it gets opened there in scaled mode. Either way, the first time anyone sees your app on a 13-inch screen should not be the person deciding whether it ships.

The failure is almost always the same one, and it's almost always at the bottom of the screen. A primary button pinned with a hardcoded offset instead of the safe area, so on iPad it sits under the home indicator, or just off the edge entirely. Your Continue button. Your Sign Up button. The one thing the reviewer needs to press to see the rest of the app, unreachable, which from their side is indistinguishable from an app that doesn't work.

Ten minutes in the simulator covers most of it: launch on the largest iPad and the smallest, rotate both to landscape, and walk the flows a reviewer walks: onboarding, sign-up, the paywall, the settings screen with the delete button in it. Open every modal and sheet. Then look specifically at the bottom hundred points of each screen, because that's where the bodies are. If you declare iPad support you'll also need real iPad screenshots in App Store Connect, and screenshots that are obviously stretched iPhone captures are their own small rejection waiting to happen.

TEST ON

Largest iPad and smallest iPad, portrait and landscape, plus one older small iPhone

LOOK AT

The bottom of every screen.
Safe areas, sheets, keyboard-avoidance, split view.

EFFORT

Ten minutes, once, and again after any layout change

03

Explain every permission like the reader has

never used your app. Because they haven't.

GUIDELINE 5.1.1 · DATA COLLECTION AND STORAGE

Every permission you request carries a purpose string, the line of text in the system dialog that you wrote months ago in a hurry and never looked at again. Mine said something like “We need access to your photos.” That is a sentence describing what I want, not why, and it reads to a reviewer as an app hoovering up data for reasons it won't say out loud.

Be more explicit than feels necessary. Name the feature, name what happens to the data, and say when. *“Photo access lets you attach an image to a job report. Photos are uploaded only to the report you choose and are never scanned or shared.”* That feels like over-explaining to you because you know what a job report is. The reviewer does not, and there is no mechanism by which they can ask you. Assume zero nuance, zero context, and zero charity, and write for that reader.

Two things that go with it. Ask for the permission at the moment the feature needs it, not in a wall of dialogs at launch. A cold-start permission gauntlet reads as a data grab and frequently gets flagged on its own. And make sure the feature the string describes is actually findable in the build you submitted. If your string promises background location for trip tracking and the reviewer can't locate trip tracking, you've handed them a reason to reject that isn't even about permissions any more.

THE ONE PEOPLE FORGET UNTIL THE DAY OF

Your App Privacy answers in App Store Connect, the nutrition label, have to match reality, including whatever your SDKs are doing. Analytics, crash reporting, ads and attribution libraries all collect things, and “I didn't know my analytics SDK collected a device identifier” is a true statement that will not help you. Go through the list one library at a time before you fill that form in, not after it gets flagged.

REWRITE

Every `NS...UsageDescription` string in your `Info.plist`. All of them, out loud.

THE TEST

Read the string to someone who's never seen the app. Can they say what feature it's for?

TIMING

At the point of use. Never a stack of prompts on first launch.

The reviewer isn't hostile and isn't on your team. They're a stranger with a queue, and everything ambiguous resolves against you.

The rest of the list, in the order they tend to bite.

None of these are obscure. All of them are things you can check tonight. The reason they still catch people is that every one of them is invisible from inside your own build.

04

Finish the app. All of it, including the boring corners.

GUIDELINE 2.1 · APP COMPLETENESS

Placeholder text, dead links, a Terms of Service link that 404s, a settings row that does nothing, a “coming soon” section, a support email that bounces. Any one of these is enough. Reviewers are specifically trained to poke at the parts of an app that look unloved, because that’s where the dishonest ones hide things. Tap every row in Settings. Follow every link, including the ones in your footer and your paywall.

The commonest version of a 2.1 rejection isn’t even in the app, though. It’s the demo account. If anything sits behind a login, App Review needs working credentials in the review notes, and they need to still work a week later. Not your personal account. Not one that gets rate-limited after three logins. A dedicated review account, with realistic data already in it, that you don’t touch. If your app needs hardware, a specific region, or a physical device to demonstrate, say so and attach a video.

05

Crashes on the paths you never walk.

GUIDELINE 2.1 · APP COMPLETENESS

Your app doesn’t crash for you because you always use it the same way: signed in, with data, on good wifi, having granted every permission the first time you were asked. The reviewer does the opposite of all four, which is why the crash they find is one you’ve genuinely never seen.

Four paths worth walking deliberately before you submit. A fresh install with an empty account, where every list is empty and something is dividing by zero. Airplane mode, and then the more evil version, a very slow connection that times out halfway. Every permission *denied*: tap “Don’t Allow” on each one and see whether the app recovers gracefully or sits on a

spinner forever, which is the one I'd bet on. And a background-and-return in the middle of a flow, especially anything with a timer, a camera or a payment in it. Run it on the newest OS beta while you're at it; review often is.

06

A privacy policy that actually loads.

GUIDELINE 5.1.1(I) · PRIVACY POLICIES

You need one in two places: a URL in App Store Connect, and a link inside the app itself that a user can reach without an account. Both get checked, and the in-app one is the one people forget. It belongs somewhere obvious: Settings, and on your sign-up screen next to the Terms.

Then click your own link from a device that isn't yours. The rejections here are almost never about the contents of the policy; they're about a link that 404s, a page behind a login, a Notion doc that isn't publicly shared, or a staging URL nobody remembered to swap. The policy itself needs to name what you collect, who you share it with, how long you keep it, and how someone deletes it, which should match, word for word, what item 01 actually does.

07

Your screenshots are a promise. Keep it.

GUIDELINE 2.3 · ACCURATE METADATA

Everything on your product page is treated as a claim about the binary you submitted. Screenshots showing a feature that's behind a feature flag. A description mentioning integrations you haven't built. A title stuffed with keywords. Marketing imagery that's mostly text and lifestyle photography rather than the actual app. Each of those is a metadata rejection, and they sting, because the app is fine. The page around it isn't.

The check is mechanical: put your screenshots on one side of the screen and the running app on the other, and confirm a stranger could find every single thing pictured. Same for the description. If a feature is real but hard to reach, tell them where it is in the review notes rather than hoping.

The paywall is the most scrutinised screen in your app.

GUIDELINE 3.1.1 · IN-APP PURCHASE · AND 3.1.2 SUBSCRIPTIONS

It's where the money is, so it's where the attention is. Three things get people. First, pricing presented to flatter rather than inform: the annual plan advertised as a monthly-looking number with the real charge in grey text underneath. If a user could be surprised by the amount that leaves their account, rewrite it. State the actual charge, the actual period, and that it renews automatically.

Second, the required furniture: a subscription paywall needs a visible **Restore Purchases** control and links to your Terms of Use and Privacy Policy, on the paywall itself, not three screens away. Third, routing digital purchases outside Apple's system. If the thing being bought is consumed inside your app, it goes through In-App Purchase, and a button that quietly opens Safari to your Stripe checkout is the version of this that gets found.

Test the whole thing in sandbox: buy, cancel, restore on a second device, and buy again. Reviewers do press restore.

“It works” and “it looks finished” are different bars.

GUIDELINE 4.0 · DESIGN

Apple rejects apps that feel unfinished even when nothing is technically broken: default system styling everywhere, inconsistent spacing, a stretched icon, text truncating at the largest accessibility size, tap targets too small to hit, no empty states, no loading states, an error that shows the raw server message to the user.

This one is subjective and people resent it for that reason, but the underlying standard is reasonable: does this look like someone cared. Two cheap wins that move the needle more than a redesign: draw a real empty state for every list, and crank the system font size to maximum and fix whatever breaks.

Don't put a login wall in front of things that don't need one.

GUIDELINE 5.1.1(II) · PERMISSIBLE USES

If a feature doesn't depend on an account, it can't require one. A recipe app that shows nothing until you sign up, a calculator gated behind an email capture, a browse-only catalogue demanding a phone number. All rejectable, and all of it reads to Apple as harvesting contact details in exchange for content you were going to give away anyway.

The fix is usually a guest mode or a browsable core with sign-in prompted at the first genuinely account-shaped action: saving, syncing, purchasing, posting. This has a second benefit, which is that your activation rate goes up. If your whole app genuinely requires an account, that's fine. Say why in the review notes and make sure the demo credentials work.

Your permission priming screen can't say "Enable Notifications."

GUIDELINE 5.1.1 · WITH A NOD TO THE HUMAN INTERFACE GUIDELINES

This one is genuinely obscure and it catches good teams. The pattern everyone uses, a custom screen before the system dialog explaining why you want the permission, is fine and encouraged. What's not fine is a button on that screen labelled "Enable Notifications" or "Allow Location," because the user taps it believing they've made the choice, and then the real system dialog appears and they're answering the same question twice. Worse, some implementations only offer the yes button, so the screen is a gate rather than an explanation.

Use a neutral label, **Continue** or **Next**, and always provide a way past without granting: "Not now," and it has to actually work. Explain freely, decide nowhere. The system dialog is the only place the decision gets made.

Be an app, not a website in a jacket.

GUIDELINE 4.2 · MINIMUM FUNCTIONALITY · AND 4.3 SPAM

A web view pointed at your existing site, with no native functionality, is not an app, and Apple has said so for years. Neither is a near-identical resubmission of something already on the store, nor five thin variants of your own app published under different names for different niches. That's the 4.3 spam rule, and it also covers "we made one app per client" business models, which surprises agencies every single time.

If a chunk of your app is genuinely a web view, that's survivable, but earn the install around it: notifications, offline behaviour, camera, share sheet, widgets, biometrics, something the browser can't do. And if you're shipping variants, consolidate them into one app with configuration, or be prepared to make the case in writing.

PART THREE · SUBMIT DAY

Write the review notes like you're handing over a job, because you are.

The review notes field is the only channel you have to a human before the decision gets made. Most people leave it blank. It is the cheapest insurance on this entire list.

Fill in the teal parts, delete what doesn't apply, and keep it short and factual. You're not selling here. You're removing every reason for someone to get stuck and write "unable to locate" in a box.

COPY THIS

DEMO ACCOUNT

Email: `review@yourdomain.com`

Password: `.....`

This account is dedicated to App Review, is pre-populated with sample data, and is not rate-limited.

WHAT THE APP DOES

[Two sentences, plain language. What it is and who it's for.]

WHERE TO FIND THINGS

- Account deletion: `Profile tab → Settings → Delete Account`
(Deletes the account and all associated data. A 30-day recovery window is disclosed in-app, after which data is permanently purged.)
- Privacy policy: in-app at `Settings → Privacy Policy`, and at `https://...`
- Subscription paywall: [where it appears, and what triggers it]
- Restore Purchases: [where the control lives]

PERMISSIONS AND WHY

- Camera: used only to attach a photo to a report. Requested at the point of use.
- Notifications: used only for reminders the user schedules themselves.

The app functions without granting any of these.

NOTES

- [Anything needing hardware, a specific region, or setup, plus a video link if so.]
- [Anything that looks unusual and has a good reason.]

Contact for anything blocking: `name, email, timezone`

Teal marks what you replace. The “where to find things” block is doing the real work. Most rejections I’ve seen in the wild are a reviewer failing to find something that was right there.

And the run-down, for the night before. If you can’t honestly tick a row, that’s tomorrow’s work rather than tonight’s submission.

THE CHECK	GUIDELINE	WHAT YOU ACTUALLY DO
01 · Delete account	5.1.1(v)	Stranger finds it in ten seconds. Soft delete with a disclosed purge. SIWA token revoked.
02 · iPad	2.1 / 4.0	Both iPad sizes, both orientations. Look at the bottom of every screen.
03 · Permission strings	5.1.1	Rewrite every one. Name the feature and the benefit. Ask at point of use.
04 · Completeness	2.1	Tap every row, follow every link. Working demo account in the notes.
05 · Crashes	2.1	Empty account, no network, all permissions denied, backgrounded mid-flow.
06 · Privacy policy	5.1.1(i)	In-app and in App Store Connect. Open both links from a device that isn't yours.
07 · Metadata	2.3	Every screenshot and claim findable in the submitted build.
08 · Paywall	3.1.1 / 3.1.2	Real price stated plainly. Restore, Terms, Privacy on the screen. Sandbox-tested.
09 · Design quality	4.0	Empty states, loading states, largest font size, no raw error strings.
10 · Login wall	5.1.1(ii)	Anything not account-dependent works without signing up.
11 · Priming screens	5.1.1 / HIG	Neutral button label, a working "Not now," decision made only in the system dialog.
12 · Originality	4.2 / 4.3	Native value beyond a web view. One app, not five variants.

A rejection isn't a judgement and it isn't the end of anything. It's a few days, a polite reply, and a resubmission. If you do get one, answer it in Resolution Center like a colleague rather than a defendant. Say what you changed, or say why the thing they're describing isn't what's happening and point them to the screen. Attach a screenshot. People on the other end are reasonable when you make it easy for them to be.

But launch week is the one week where days actually cost you something, and every item above takes minutes tonight. That trade has never once looked close to me.

ONE MORE THING

If you got rejected and the reason doesn't make sense.

Apple's rejection text is a guideline number and a paragraph of boilerplate. Translating it into the specific thing in your specific app is the hard part, and it's hard mostly because you're too close to it.

Send me the rejection message and a sentence about what your app does. I'll tell you what I think they're actually looking at and how I'd respond. No code and nothing proprietary needed. The guideline number and their wording is usually enough.

GET IN TOUCH

Email hi@davecto.com with the subject line **"App Store Checklist."**

More guides like this one, for people trying to ship software without embarrassing themselves.

Weekly, plain-language breakdowns on Instagram.

[@davecto](#)

A note on sourcing: guideline numbers are from Apple's App Store Review Guidelines as of writing, and that document gets renumbered and rewritten more often than people expect, so check the current text before you quote a number back at anyone. The 24-hour review figure is Apple's own published claim about typical turnaround, not a promise, and first

submissions from new accounts routinely take longer. Items 01 and 03 come from rejections on my own apps; the rest are drawn from the rejection reasons that recur most often across developer write-ups and forums, filtered through what I've seen myself, which makes them a well-founded ordering rather than a measured one. Everything here is about the public App Store on iOS and iPadOS. Enterprise distribution, TestFlight and the alternative-marketplace rules in the EU work differently. I have no relationship with Apple beyond being a developer who has argued with them.