ITCH.STUDIO

Senior iOS engineering · care plans · AI features

App Health Check

Pacelark: Running Club Companion (iOS)

Prepared for The Founder, Pacelark Ltd

Prepared by John Durcan · itch.studio

App version reviewed v2.4.1 (build 187), live on the App Store

Review period 2–6 November 2026

Report date 9 November 2026

SAMPLE REPORT

Pacelark and Pacelark Ltd are fictional, and every finding, figure and screen in this report is invented for illustration. It shows the structure, depth and honesty you can expect from an itch.studio App Health Check: a fixed-price, one-week audit of your live iOS app. Your own report stays confidential.

Executive summary

Pacelark is a well-loved product with a solid core: a 4.6-star rating, an engaged user base, and a codebase that is better organised than most apps of its age. Nothing here is a crisis. But the app has not shipped an update in 14 months, and that gap has quietly accumulated risk in three places that now need dates attached to them.

First, the app was last built against the iOS 18 SDK with Xcode 16. Since 28 April 2026 Apple has required every upload to App Store Connect to be built with Xcode 26 and the iOS 26 SDK, which means the very next update, even a one-line bug fix, must first be migrated to current tooling. That migration is where several deprecated APIs (notably in location tracking and background processing, the heart of a running app) will stop being warnings and start being work. The current version keeps working in the meantime, but left longer, this lands at the moment you most need an urgent fix instead of as planned work.

Second, stability is worse than your rating suggests. Crash reporting shows a 99.1% crash-free sessions rate (I would want to see at least 99.5%), with two crash clusters accounting for 70% of the total, both fixable. Your recent reviews mention crashes and GPS drop-outs more often than any other complaint; the star rating is being carried by loyal club users, not by stability.

Third, compliance: the updated age-rating questions Apple has required since January 2026 are unanswered, and the account-deletion flow ends in an email to support, either of which can hold up your next submission. Your subscription paywall also needs its terms made clearer. None of this touches the version live today.

The good news: the fix list is finite, sequenced below, and roughly 69–84 hours of senior work spread over a quarter. Section 8 sets out exactly what I would do, in what order, and what I would explicitly not spend money on.

Scorecard

Area	Status	One-line summary
1. Platform & SDK currency	AT RISK	Built on iOS 18 SDK / Xcode 16; Apple has required Xcode 26 since April 2026. Blocks all future updates until migrated.
2. Stability & performance	NEEDS WORK	99.1% crash-free; two clusters cause 70% of crashes. Launch time acceptable, memory fine.
3. App Store compliance	AT RISK	Age-rating answers outstanding; account deletion ends in an email; paywall terms unclear.
4. Accessibility	NEEDS WORK	VoiceOver mostly unlabelled on the run screen; Dynamic Type breaks two key layouts.
5. UX & App Store presence	NEEDS WORK	Strong retention; onboarding drop-off at permissions; screenshots and metadata two years old.
6. Security & data handling	HEALTHY	Sensible auth, TLS everywhere, no secrets in the binary. Two low-severity items.

1. Platform & SDK currency

AT RISK

Where the app stands against Apple's moving baseline: build tooling, SDK requirements, deprecated APIs, and OS-version support.

1.1 App is built on an SDK Apple no longer accepts for uploads [HIGH]

Pacelark v2.4.1 was built with Xcode 16 against the iOS 18 SDK (September 2025), which met Apple's rules at the time. Since 28 April 2026 every upload must be built with Xcode 26 or later and the iOS 26 SDK, so the next update of any kind requires migrating the project first. On inspection, the project builds under Xcode 26 with 3 errors and 41 deprecation warnings: a real but modest migration, best done deliberately rather than under deadline pressure.

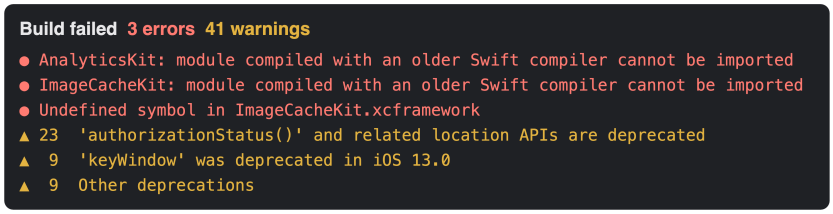
1.2 Deprecated location and background APIs at the core of the product [HIGH]

Run tracking uses location and background-task patterns that are formally deprecated in the current SDK. They still function today, but this is the app's single most important code path, and each OS release now carries a real chance of degraded behaviour, consistent with the GPS drop-out reviews in Section 5. Migrating to the current live-tracking APIs also unlocks Live Activities on the lock screen, which for a running app is a visible product win, not just plumbing.

1.3 Minimum deployment target can rise without losing users [LOW]

The app supports iOS 15+, but your analytics show 97.8% of active users on iOS 17 or later. Raising the minimum to iOS 17 sheds legacy code paths and reduces every future estimate. The 2.2% on older versions keep the current app version; they do not lose anything they have today.

Figure 1: project built under Xcode 26 ILLUSTRATIVE MOCK-UP



1.1 A short list. All three errors come from two out-of-date SDKs, which is why this is routine work if planned.

What this means for you: Nothing new can ship until the toolchain migration happens, so it is item one in the action plan. The live app keeps working meanwhile, but done this quarter the migration is routine; left until you need an urgent fix, it becomes an emergency.

2. Stability & performance

NEEDS WORK

Crash data from your existing reporting service (30-day window), plus profiling of launch, memory and energy use on device.

2.1 Two crash clusters cause 70% of all crashes [HIGH]

Crash-free sessions sit at 99.1%, below the 99.5% or better I would want to see. Cluster A (a race condition when a run is ended while the app is backgrounded) accounts for 47% of crashes and loses the user's recorded run: your angriest review category. Cluster B (an unhandled failure when club photos exceed a memory threshold on older devices) accounts for 23%. Both are well-understood patterns with clear fixes; together they are roughly 8–10 hours of work including regression tests.

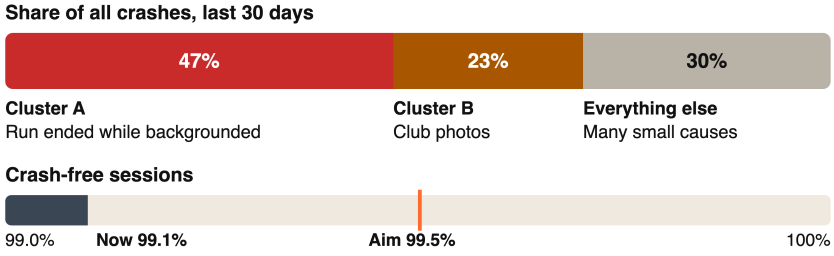
2.2 Cold launch is acceptable, with one cheap win [LOW]

Cold launch measured at 1.9s on an iPhone 12, inside Apple's guidance, though club-feed images are fetched before first paint and account for ~0.5s of that. Deferring the feed fetch until after first render is a small change with a visible snappiness gain.

2.3 Battery use during runs is fine; keep it that way [INFO]

Energy profiling during a simulated 45-minute run showed sensible GPS duty-cycling. The API migration in 1.2 must preserve this; battery regressions are the highest-churn mistake in running apps, so the action plan includes an energy re-test as part of that migration, not as an afterthought.

Figure 2: where the crashes come from ILLUSTRATIVE MOCK-UP



2.1 Fixing two bugs removes 70% of all crashes and lifts the crash-free rate. Cluster A is the one that loses a recorded run.

What this means for you: Fixing two specific bugs removes the majority of your crashes and directly targets the complaint driving your worst reviews. This is the highest ratio of user-visible improvement per hour anywhere in this report.

3. App Store compliance

AT RISK

Checks against current App Review Guidelines, privacy requirements, and subscription rules: the items that can hold up a submission.

3.1 Updated age-rating questions unanswered; two SDKs out of date [HIGH]

Apple updated its age-rating system and has asked every app to answer the new questions, by 31 January 2026, to avoid an interruption when submitting updates. Pacelark's answers are outstanding, which is exactly the kind of finding better handled now than discovered mid-release. Separately, two third-party SDKs in the build (analytics and image caching) are well behind their current versions; updating them, and checking their privacy declarations, is part of the toolchain migration.

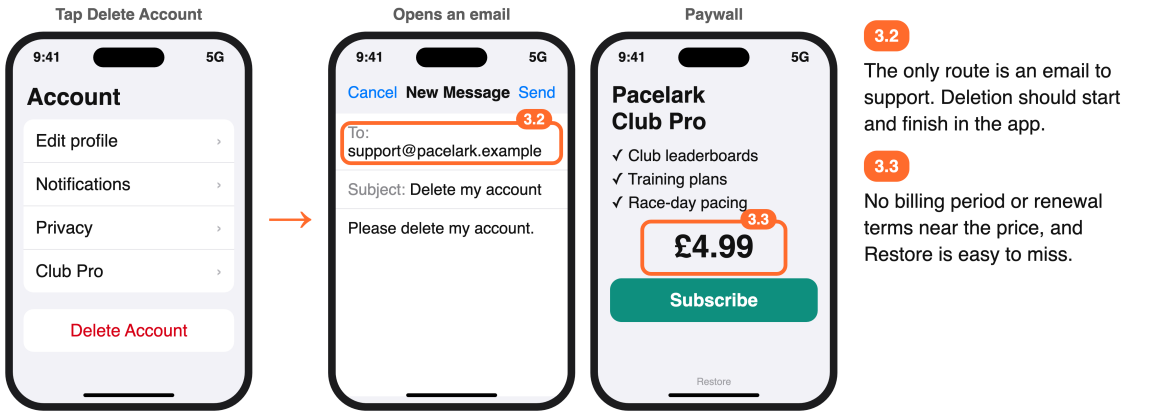
3.2 Account deletion ends in an email to support [HIGH]

Apps offering account creation must let people start deleting their account in the app, and apps outside highly regulated industries should not make them finish by email or phone. Pacelark's Delete Account option only opens an email to support. It got through review before, but it could be picked up at any future submission. A straightforward build: roughly 6–8 hours including the backend call you already expose in your web dashboard.

3.3 Subscription paywall does not show its terms clearly [MEDIUM]

The Pacelark Club Pro paywall does not show per-period pricing and renewal terms clearly enough for App Review's subscription rules, and the restore-purchases path is buried. Beyond compliance, your paywall is built on the older StoreKit API; migrating it improves receipt handling and family-sharing behaviour, and gives you win-back offers you currently cannot run.

Figure 3: account deletion and paywall ILLUSTRATIVE MOCK-UP



What this means for you: Two of these three could hold up your next submission, but all three fold neatly into the same release as the toolchain migration: one submission that clears the compliance debt in full.

4. Accessibility

NEEDS WORK

VoiceOver, Dynamic Type, contrast and touch-target review of the six core screens.

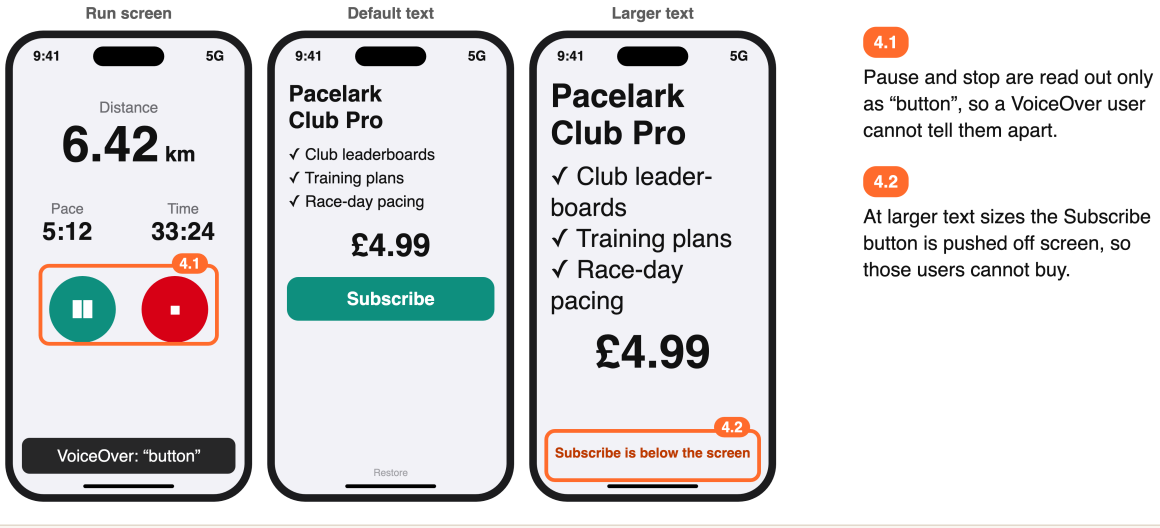
4.1 The run screen is effectively unusable with VoiceOver [MEDIUM]

Pace, distance and the pause/stop controls have no accessibility labels; VoiceOver reads "button, button". For a fitness app this is a real inclusion failure. Labelling the core screens is ~4 hours; I would also add an accessibility statement to your site once done, which club organisers increasingly ask for.

4.2 Dynamic Type breaks two key layouts [MEDIUM]

At larger text sizes the leaderboard truncates member names entirely, and the paywall's purchase button is pushed off-screen, meaning users with large text literally cannot subscribe. The paywall rebuild in 3.3 should simply be built text-size-correct rather than patched.

Figure 4: VoiceOver and larger text sizes ILLUSTRATIVE MOCK-UP



What this means for you: The paywall issue in particular means anyone using large text cannot subscribe, so accessibility here is revenue work, not box-ticking.

5. UX & App Store presence

NEEDS WORK

Review mining (last 12 months, UK storefront), onboarding funnel, and listing quality.

5.1 Your reviews tell you exactly what to fix, and it matches this report [INFO]

Of 214 written reviews in the last year: 31% mention GPS drop-outs or lost runs (Sections 1.2 and 2.1), 18% mention crashes (2.1), and 12% ask for lock-screen/watch visibility during runs (the Live Activities win in 1.2). The complaints cluster almost perfectly around this report's two highest-priority engineering items: useful confirmation the plan targets what users actually feel.

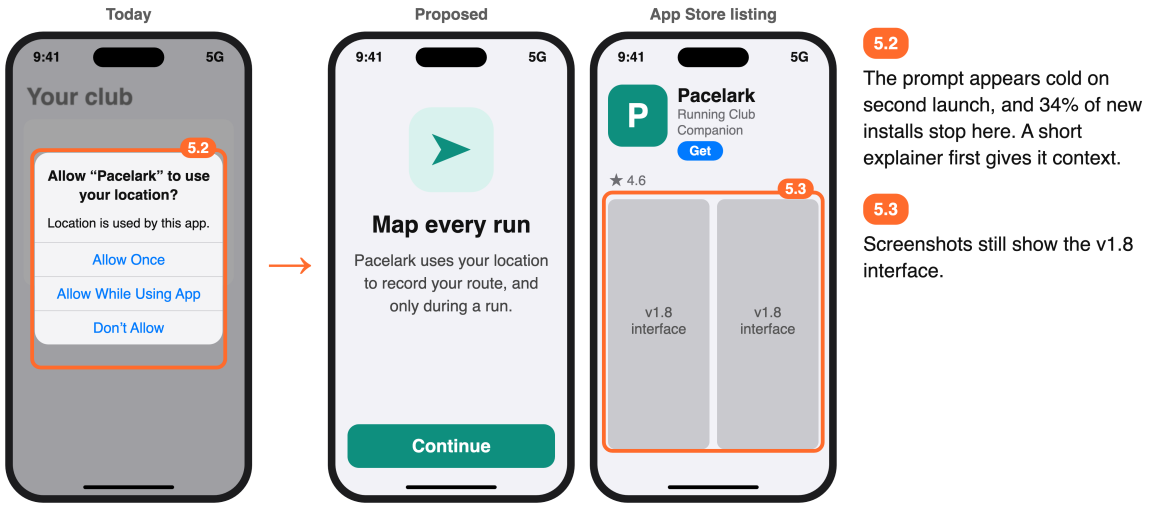
5.2 Onboarding loses a third of new users at the permissions step [MEDIUM]

Analytics show 34% of new installs abandon at the location-permission prompt, which currently appears cold on second launch with no explanation. Adding a single pre-permission screen explaining why a running app needs location ("Always" only during runs) usually recovers part of this drop-off. Two hours of work; measurable within a fortnight.

5.3 The listing is two years old [LOW]

Screenshots show the v1.8 interface, the description leads with features you have since surpassed, and you have no promotional text for seasonal moments (spring marathons, January). A refreshed listing is half a day and can lift conversion, worth doing in the same release as the update, when "What's New" is genuinely new.

Figure 5: location permission and App Store listing ILLUSTRATIVE MOCK-UP



What this means for you: The permissions fix in 5.2 is the single cheapest growth lever in this report: a third of your new installs are currently evaporating at one screen.

6. Security & data handling

HEALTHY

Static review of the binary and configuration, transport security, credential storage, and third-party data flows. This was not a penetration test.

6.1 Fundamentals are sound [INFO]

Transport security is properly enforced, tokens live in the Keychain, no API secrets ship in the binary, and sign-in uses modern flows including Sign in with Apple. Whoever set this up did it properly.

6.2 Two low-severity items [LOW]

Run routes are cached unencrypted in the app container: fine for App Store rules, but location history is sensitive; enabling file protection is an hour’s work. And the analytics SDK is configured to collect precise location where coarse would serve, which is worth tightening before your next privacy-label update rather than after a user asks about it.

> **What this means for you:** Nothing urgent. The two items above are hygiene, folded into the main release at negligible cost.

7. Prioritised action plan

Everything above, sequenced. Effort is senior-engineer hours, honest rather than optimistic. You own this list; any competent iOS developer can execute it.

Ref	Action	Severity	Effort	When
1.1	Migrate project to current Xcode/SDK; resolve errors and deprecation debt; update the two out-of-date SDKs (unblocks everything below)	**HIGH**	12–16 h	Month 1
3.1	Answer the updated age-rating questions; confirm privacy declarations after the SDK updates	**HIGH**	3–4 h	Month 1
2.1	Fix the two crash clusters (run-end race; club-photo memory) with regression tests	**HIGH**	8–10 h	Month 1
3.2	Build in-app account deletion	**HIGH**	6–8 h	Month 1–2
1.2	Migrate location/background tracking to current APIs; add Live Activities; energy re-test	**HIGH**	14–18 h	Month 2
3.3 + 4.2	Rebuild paywall on modern StoreKit, compliant and Dynamic Type-correct	**MEDIUM**	10–12 h	Month 2–3
5.2	Pre-permission explainer screen in onboarding	**MEDIUM**	2 h	Month 1 (quick win)
4.1	VoiceOver labels on core screens	**MEDIUM**	4 h	Month 2
6.2	Encrypt route cache; coarse-location analytics config	**LOW**	2 h	Month 2
2.2	Defer feed fetch to post-first-paint	**LOW**	2 h	Month 3
5.3	Refresh App Store listing alongside the release	**LOW**	4 h	Month 3
1.3	Raise deployment target to iOS 17	**LOW**	2 h	Month 3

Total: roughly 69–84 hours across a quarter. Months 1–2 clear the two items most likely to hold up review and the crash clusters; month 3 is polish and growth.

8. My recommendation, and what I would not do

If I were looking after Pacelark, I would run the plan above as three monthly releases on a Product Care plan (£2,200/month, 12 hours of changes) topped up with a one-off fixed quote for the toolchain migration and tracking-API work, since those exceed a single month's allowance. That combination clears the entire list this quarter and then drops to steady-state care.

What you do not need: you do not need the Studio Partner tier; Pacelark's cadence does not justify 24 hours a month once this backlog clears. You do not need a redesign; the interface tests well and your users like it. And I would not build the Android version some of your reviews request until the iOS foundation is current; adding a second platform on top of this backlog would double the cost of every item in this report.

Equally valid alternatives: hand this report to a contractor for the quarter and take Essential Care (£950/month) afterwards for compliance-only upkeep; or if budget is the constraint, do items 1.1, 3.1, 3.2, 2.1 and 5.2 only (roughly 31–40 hours), which clears what could hold up your next release and the worst crashes, and defer the rest. I am happy to talk through any of the three.

Whatever you choose, the one thing I would not do is nothing: the SDK gap is the only item in this report that gets more expensive every month it waits.

Appendix: scope, method and tools

This Health Check covered: a full build of the current codebase under current tooling; static review of project configuration, dependencies and binary; 30 days of crash and analytics data from your existing services; on-device profiling (launch, memory, energy) on iPhone 12 and iPhone 16 hardware; App Review Guideline and privacy-requirement checks against the current published rules; a VoiceOver, Dynamic Type and contrast pass over the six core screens; and a reading of all 214 written UK reviews from the last 12 months.

It did not include: penetration testing, backend/API security review, load testing, or legal advice; where findings touch legal territory (privacy, accessibility) they flag engineering reality, and your counsel should own the legal position.

Tools: Xcode Instruments, current Xcode toolchain build, your crash-reporting and analytics dashboards (read-only access), and App Store Connect metadata review. Screens in a real report come from a test account, never a real user’s data.

About itch.studio

itch.studio is John Durcan: a senior iOS engineer, writing production Swift since 2019 and SwiftUI since its first release, who has designed, built and shipped his own apps end to end, after twelve years running a marketing agency of more than twenty people. Every Health Check ends the way Section 8 does: with a specific recommendation, including "you don’t need me" when that’s true.

hello@itch.studio · itch.studio · **Book a call or a Health Check at itch.studio**