

Testing checklist: wearable companion app

Source: <https://howtotestaproduct.com/how-to-test/wearable-companion-app/>

Generated: 2026-10-07. Licence: CC BY 4.0.

☐ 1. release smoke tests

Done when: On the release build, the app opens to a demo mode without a band, pairs with a real band, and shows a reading synced in the last minute.

Risks: store-rejection. Tool class: mobile automation. Usually run by: QA engineer.

☐ 2. end-to-end functional tests

Done when: A threshold breach recorded on the band produces a phone notification, and a band disconnected for longer than the set interval shows a 'not connected' warning instead of the last value.

Risks: user-harm. Tool class: mobile automation. Usually run by: QA engineer.

☐ 3. exploratory testing

Done when: A session log covers walking out of range, toggling Bluetooth, force-quitting the app, low band battery and a second phone sending repeated pairing requests within range, and states what the user saw in each case and whether band alerts kept working.

Risks: user-harm. Tool class: test management. Usually run by: QA engineer.

☐ 4. compliance checks

Done when: A signed review lists every health alert and claim in the app and store listing, and states for each whether it makes the product a medical device and who approved it.

Risks: regulatory-penalty. Tool class: test management. Usually run by: product manager.

☐ 5. security scanning

Done when: The scan of the app binary and its sync API shows no open high-severity finding and no health data stored unencrypted on the phone.

Risks: data-breach. Tool class: SAST scanner. Usually run by: security specialist.

☐ 6. penetration testing

Done when: A tester with a second phone in range cannot pair with, read data from or send commands to a band already paired to another account.

Risks: data-breach. Tool class: proxy. Usually run by: security specialist.

☐ 7. analytics and event tracking tests

Done when: A network capture during pairing, sync and browsing shows no heart rate, sleep or cycle data sent to an analytics or advertising endpoint the user did not authorise.

Risks: data-breach. Tool class: proxy. Usually run by: security specialist.

☐ 8. resilience and failure testing

Done when: With the cloud API blocked, the app still shows readings stored on the band and phone, and queued data uploads once when the cloud returns.

Risks: dependency-failure. Tool class: chaos tool. Usually run by: developer.

☐ 9. device and browser compatibility

Done when: Pairing, background sync and alerts work on each phone model and OS version in the support list, with battery-saver mode both on and off.

Risks: store-rejection. Tool class: device cloud. Usually run by: QA engineer.