

Testing checklist: mobile game

Source: <https://howtotestaproduct.com/how-to-test/mobile-game/>

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

☐ 1. release smoke tests

Done when: On the release build, a new player completes the tutorial, buys one item with a sandbox account and finds it in the inventory after a restart, and the loot box screen shows item odds before the buy button.

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

☐ 2. end-to-end functional tests

Done when: A new account set to under 13 starts with voice and text chat off, and only a verified parent can turn them on.

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

☐ 3. exploratory testing

Done when: A tester acting as a child tries to buy currency and chat with strangers, and each attempt stops at a parental gate.

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

☐ 4. data integrity and migration testing

Done when: For a sample of sandbox purchases, every charge in the store's purchase records matches a purchase the player confirmed in the game, and each purchased item appears in the player's inventory once.

Risks: money-loss. Tool class: data validation. Usually run by: developer.

☐ 5. payment and money-flow testing

Done when: No sandbox purchase can start from a loading screen, sleep wake or item preview, and every purchase asks for a confirmation that names the item and price.

Risks: money-loss. Tool class: mobile automation. Usually run by: QA engineer.

☐ 6. compliance checks

Done when: A reviewer confirms each loot box shows item odds before purchase, and a player under 13 cannot reach chat or purchases without a parent's consent.

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

☐ 7. performance and load testing

Done when: A load test at the launch or event forecast plus the margin the team set keeps login and match start within the team's target.

Risks: outage-under-load. Tool class: load generator. Usually run by: developer.

☐ 8. resilience and failure testing

Done when: When the service-discovery cluster or session store degrades, players see a queue screen and keep their inventory and progress after recovery.

Risks: outage-under-load. Tool class: chaos tool. Usually run by: developer.