

Testing checklist: neobank

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

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

☐ 1. release smoke tests

Done when: On every store build, a tester signs up a new test applicant, passes the ID step, receives account details and makes a first card payment in the sandbox.

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

☐ 2. end-to-end functional tests

Done when: An applicant with an implausible address or mismatched selfie is sent to manual review, and a closed test account's balance arrives at the external bank account.

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

☐ 3. penetration testing

Done when: An external tester cannot download another applicant's ID photo or selfie, and cannot reuse a passed ID check to open a second account.

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

☐ 4. data integrity and migration testing

Done when: After each partner bank statement, the sum of customer balances in the app ledger equals the pooled account statement; a run that finds a difference lists it by customer.

Risks: data-corruption, money-loss. Tool class: data validation. Usually run by: QA engineer.

☐ 5. compliance checks

Done when: Test applicants whose names are on the current sanctions list are stopped at sign-up, and the screening log shows the list version used.

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

☐ 6. regression suite

Done when: Every release reruns the sign-up screening and ledger posting cases, and none changes outcome against the previous release.

Risks: regulatory-penalty, data-corruption. Tool class: API client. Usually run by: QA engineer.

☐ 7. accessibility testing

Done when: A screen reader user completes sign-up, including the ID and selfie step or its stated alternative, without sighted help.

Risks: accessibility-liability. Tool class: accessibility checker. Usually run by: accessibility specialist.

☐ 8. integration tests

Done when: A card payment, a refund and an incoming transfer made in the sandbox each appear once in both the app ledger and the partner bank feed.

Risks: dependency-failure, data-corruption. Tool class: API client. Usually run by: developer.

☐ 9. security scanning

Done when: Each release build is scanned and no open high finding remains on stored ID images, tokens or card details on the device.

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

☐ 10. resilience and failure testing

Done when: With the partner bank feed stopped, the app marks balances as delayed instead of showing stale figures as current, and blocks withdrawals it cannot confirm.

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