Prioritised Remediation Plan — Public Redacted Copy

Final Whistle Card Collection — Prioritised Remediation Plan

Working document · 17 October 2026

This plan summarises open actions identified by static review of the uploaded project and the DPIA. It is not a penetration-test report, a declaration of vulnerability, or proof of deployment state. No production database or real payment actions were used.

P0 — required before release sign-off

■ Test age gates and parental permissions at the UI and all relevant server endpoints, including direct requests, missing parent links, revoked permissions and feature-specific approvals. Use synthetic staging accounts.

■ Confirm deployed Firestore rules and Netlify function versions match the reviewed source; test cross-account reads/writes in a separate Firebase project.

■ Verify the relevant deployed security controls and configuration against the reviewed source in a controlled test environment. Detailed component names and test steps are retained internally.

P1 — high-priority operational and security controls

■ Verify checkout consent and terms version against server-controlled current values; test ineligible users without creating a real payment.

■ Test Stripe webhook signatures, duplicate/out-of-order events, cancellation, renewal and failed-payment handling in test mode.

■ Verify server-side scanner limits under concurrency and ensure images/full prompts are not written to logs; mock upstream AI responses.

■ Test account deletion and injected failure paths with synthetic accounts, including linked-child restrictions and subscription handling.

■ Verify the relevant deployed security controls and configuration against the reviewed source in a controlled test environment. Detailed component names and test steps are retained internally.

■ Verify the relevant deployed security controls and configuration against the reviewed source in a controlled test environment. Detailed component names and test steps are retained internally.

P2 — documentation and operational readiness

■ Verify provider DPAs, subprocessors, processing locations and applicable international-transfer mechanisms.

■ Check the privacy summary and public privacy policy against actual live data flows and fields.

■ Rehearse a synthetic data-rights request and deletion request; rehearse the breach response procedure.

■ For every action, record owner, due date, build/version, test steps, expected and actual result, evidence location and residual risk.

■ Obtain controller sign-off only after the evidence is reviewed; assess ICO prior consultation if a high residual risk cannot be mitigated.

Evidence discipline

A source-code observation is not proof that a control is deployed or works. Mark an item complete only when dated evidence supports the result. Do not use production user data for destructive or access-control testing.

Next work package — deployed-version and access-control evidence

This is the next practical work package. It is deliberately limited to verification and non-production tests; it does not authorise changes to live data. Start by filling in `Deployment-Parity-Evidence-Record.md`. Current deployment parity is unverified; do not mark it matched without reproducible evidence.

A. Capture what is actually deployed

■ Record the Netlify site name, deploy ID, deploy timestamp, and the commit or uploaded build used for the live deploy.

■ Verify the relevant deployed security controls and configuration against the reviewed source in a controlled test environment. Detailed component names and test steps are retained internally.

■ Record the Firebase project ID and deployed Firestore Rules version/release date. Do not paste secret values into the evidence log.

■ Record the names (not values) of required environment variables and whether each is configured. Keep credentials and tokens out of screenshots and logs.

■ If the deployed version cannot be established, mark deployment parity **unverified** and do not infer that the reviewed code is live.

B. Non-production access-control test cases

Use a separate Firebase project and synthetic accounts: an adult account, a child account with approval pending, an approved child account, an unrelated parent, and an unrelated ordinary user. Use Stripe test mode and mocked AI calls where relevant.

• ID • Test • Expected result • Evidence to retain •

• --- • --- • --- • --- •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

• Detailed test identifiers and implementation-specific references withheld from public copy •

C. Evidence record template

Copy one record per test/action:

• Action ID: [e.g. DEP-01]

• Owner: [role/person]

• Date/time and timezone: [ ]

• Environment: [staging/test only]

• Build/deploy ID: [ ]

• Preconditions and synthetic account roles: [ ]

• Steps performed: [ ]

• Expected result: [ ]

• Actual result: [ ]

• Evidence filename/location (redacted): [ ]

• Pass / Fail / Blocked / Not run: [ ]

• Follow-up ticket and target date: [ ]

• Residual risk after result: [ ]

Do not mark a test as passed based only on the source code. If no staging environment is available, record the test as blocked and arrange a safe test environment before attempting any destructive or cross-account test.