Security Review

AI Retirement Income Planner — Version 7 · Prepared 4 June 2026 · Rev. 3 (all recommendations closed) · Audience: CIO / Information Security

Summary. The application is a single, self-contained HTML file that runs entirely in the end-user's browser. It has **no backend, no server, no user accounts, and no database**, which eliminates an entire class of server-side risk. A user's financial plan is stored only in their own browser and is never transmitted to the vendor. The principal data-egress path is **optional, user-initiated AI features**, which send the plan's figures to Anthropic's API under the **user's own API key**. No critical or high-severity issues were identified. The notable items were a third-party CDN dependency without integrity pinning (supply-chain) and the inherent client-side storage of the user's API key; **the supply-chain item has since been remediated and the others mitigated** — see the Remediation Status below and the updated findings table.

Remediation status (4 June 2026). Following this review the development team implemented and verified: (1) **F-2 — Resolved:** the unused html2canvas library was removed, and Chart.js v4.4.1 was **inlined into the file after verifying its SHA-512 hash against the published CDN integrity value** — eliminating the runtime CDN script dependency (the application is now self-contained for charting and offline-capable). (2) **F-4 — Resolved:** the interface now explicitly discloses that plan figures are sent to Anthropic when AI features are used. (3) **F-3 — Mitigated:** a shared-device caution was added (a "Remove key" control already existed). (4) **Recommendation 5 — Implemented:** a Content-Security-Policy was added with a restrictive `connect-src` whitelist (Anthropic API + exchange-rate API only), constraining data egress even under injection. (5) **Remaining low-priority items also closed:** the theme web fonts are now **self-hosted (inlined as base64)** — removing the last third-party runtime asset, so the application fetches **no external code, scripts, or fonts**; and an optional **session-only key mode** ("don't save on this device" → cleared when the browser closes) was added for shared computers. All changes were verified with no console or CSP violations. The only off-device network path that remains is the **optional, user-initiated AI call** to Anthropic (under the user's own key) and the anonymous exchange-rate lookup. Residual accepted item: client-side key storage (inherent to a keyless local app; now with a session-only option).

1. Architecture & data flow

- **Delivery:** one static `.html` file (HTML + CSS + JavaScript). No build step, no installer, no runtime server. Opened directly from disk (`file://`) or any static host.
- **Execution:** 100% client-side in the browser. All retirement calculations run locally in JavaScript.
- **State:** the entire plan lives in an in-memory object and is persisted to the browser's `localStorage` (key `retirement_planner_v5`). Users may also manually export/import a plan as a JSON file.
- **No server-side attack surface:** there is no application server, API gateway, auth service, or database to attack, patch, or breach.

2. Data handling & privacy

- **What data exists:** user-entered financial assumptions — account balances, withdrawal amounts, ages, tax parameters, pension/Social Security figures. The app does **not** request name, address, SSN, account numbers, or credentials, and does not connect to any financial institution.
- **Where it lives:** only in the user's browser (`localStorage`) and any JSON file the user chooses to export. Nothing is uploaded to the vendor.
- **Egress (the one path off-device):** when the user opts into AI features, a plain-text snapshot of the plan's figures is sent to `api.anthropic.com` over HTTPS, authenticated with the **user's own Anthropic API key**. This is user-initiated, key-gated, and governed by Anthropic's data-use policies under the user's own account. The figures are financial but not inherently identifying unless the user typed identifying text into a free-text field (e.g. a plan label).
- **Rule-based features need no network:** the calculations, charts, Plan-Health checks, Monte Carlo, stress test, historical backtest and the local "insights" all run offline with no API key and no data transmission.

3. Third-party network calls & dependencies

Endpoint / resource	Purpose	Data sent	Notes
`api.anthropic.com/v1/messages` and `/v1/models`	Optional AI chat, plan suggestions, tax-rate lookup, help-content audit, model discovery	Plan figures (financial); the user's API key in the `x-api-key` header	HTTPS; user-initiated; key-gated; uses Anthropic's documented browser-access header.
`open.er-api.com/v6/latest/USD`	Currency exchange rates for display conversion	**None** (anonymous GET; no auth, no user data)	HTTPS; 6-second timeout; failure degrades gracefully.
~~`cdnjs.cloudflare.com` — Chart.js, html2canvas~~ **(removed)**	Charting / report images	None	**Remediated:** html2canvas removed (it was unused); Chart.js v4.4.1 now **inlined** into the file (SHA-512 verified against the CDN hash) — no runtime CDN script fetch. See F-2.
~~`fonts.googleapis.com` / `fonts.gstatic.com`~~ **(removed)**	Typography	None	**Remediated:** theme fonts are now self-hosted (inlined as base64 woff2). No runtime font fetch; CSP tightened to drop the Google hosts.
`youtube.com`	Optional explainer videos	None in-app	Opened in a new browser tab only on explicit user click; no embedded player or tracking in the app.

*Note: after remediation the app fetches **no third-party assets at runtime** — Chart.js is inlined, html2canvas removed, and the theme fonts are self-hosted (inlined). The only off-device network paths are the optional, user-initiated AI call to Anthropic (under the user's own key) and the anonymous exchange-rate lookup.*

4. Secrets handling

- The only secret is the user's **Anthropic API key**, which the user pastes in once. It is stored in browser `localStorage` and sent only to `api.anthropic.com` over HTTPS.
- It is never transmitted to the vendor or any third party, and is not embedded in the file.
- As with any client-side credential, it is readable by browser dev-tools and by any script running on the same origin (see Finding F-3).

5. Attack-surface analysis

- **Cross-site scripting (XSS):** user-supplied text (plan labels, acknowledgement/dismissal notes, subtitle, AI-proposal reasons) is consistently HTML-escaped before being inserted into the DOM; the editable subtitle is handled via `textContent`. No XSS vector identified.
- **Code injection:** the application contains **no `eval()` and no `new Function()`**. AI responses that propose plan changes are parsed by a constrained handler that only writes **numeric values to named fields** — it cannot execute arbitrary code. Proposed changes are simulated and require explicit user approval ("Apply"), with one-click Undo.
- **Untrusted plan import:** imported/shared JSON is parsed (not evaluated) and passed through a sanitizer that coerces non-finite numbers to safe defaults and normalizes lump-sum records. JavaScript object-spread is not vulnerable to prototype pollution from JSON keys. No code-execution path via import.
- **Prompt injection:** a malicious AI response is bounded — its worst case is a poor numeric suggestion, which is verified against the local simulator and must be user-approved. It cannot run code or exfiltrate data.
- **Transport:** all network calls use HTTPS.

6. Findings & residual risk

ID	Severity	Finding	Recommendation
F-1	**None / Positive**	No backend, no accounts, no database; user financial data stays on-device; consistent output escaping; no `eval`; HTTPS throughout.	Maintain these properties as design invariants in future changes.
F-2	**RESOLVED** (was Low–Medium)	**Supply chain:** Chart.js and html2canvas were loaded from a public CDN (cdnjs) without Subresource Integrity. A CDN compromise could have served altered script.	**Fixed (4 Jun 2026):** html2canvas removed (unused); Chart.js v4.4.1 **inlined after SHA-512 verification** against the published CDN hash. No runtime third-party JavaScript remains.
F-3	**Low** (mitigated)	**API key at rest:** the user's Anthropic key is stored in plaintext in `localStorage` (inherent to any keyless client-side app). Readable on a shared/compromised device.	**Mitigated:** a "Remove key" control exists, an in-app shared-device caution was added, and an optional **session-only key mode** ("don't save on this device" → sessionStorage, cleared when the browser closes) was added. Residual risk accepted as inherent to a local, keyless design.

F-4	**RESOLVED** (was Low)	**AI data-egress awareness:** users may not have realized plan figures are sent to Anthropic when AI features are used.	**Fixed (4 Jun 2026):** the UI now states explicitly that plan figures are sent to Anthropic (under the user's own account/policies) when AI features run. Remains optional and key-gated.
F-5	**Informational**	When served behind Cloudflare, an email-obfuscation script (`/cdn-cgi/…/email-decode.min.js`) is auto-injected by the CDN. This is a hosting artifact, not part of the application, and is inert when run locally.	No action needed; it appears only on Cloudflare-served copies and is permitted by CSP as a same-origin path.
F-6	**Added control**	**Defense-in-depth:** no Content-Security-Policy was previously present.	**Implemented (4 Jun 2026):** a CSP `<meta>` was added with a restrictive `connect-src` whitelist (Anthropic API + exchange-rate API only), so injected content cannot exfiltrate data to other hosts. (`'unsafe-inline'` is required by the single-file inline-script design, so CSP's anti-XSS value is partial; the egress restriction is the primary benefit.)

7. Recommendations (status)

- [DONE] **1. Remove the CDN supply-chain dependency (F-2).** html2canvas removed; Chart.js inlined after SHA-512 verification. The file is now self-contained for charting.
- [DONE] **2. Reinforce key-handling guidance (F-3).** "Remove key" control (pre-existing) plus a new shared-device caution. Optional future: a session-only key mode.
- [DONE] **3. Disclose AI data egress (F-4).** The UI now states plan figures are sent to Anthropic when AI features are used.
- [DONE] **4. Preserve the security invariants (F-1)** — no backend, output escaping, no `eval`, HTTPS — now documented as explicit guardrails in the file's header comments.
- [DONE] **5. Content-Security-Policy.** Added with a restrictive `connect-src` whitelist; verified with no console/CSP violations.
- [DONE] **6. Self-hosted the web fonts** (inlined as base64) — removes the last remote asset; the app now fetches no third-party code or fonts at runtime, and the CSP no longer needs the Google hosts.
- [DONE] **7. Added an optional session-only key mode** — a "don't save on this device" choice that keeps the key only in `sessionStorage` for the browser session (for shared/public computers).

8. Scope, methodology & limitations

This review was a source-level examination of the single application file: data-flow tracing, enumeration of all network calls and third-party dependencies, secrets handling, and an output-escaping / injection review of the calculation and rendering paths and the recently added features. It was a manual code review, **not** a penetration test, dynamic scan, or formal third-party audit, and it does not assess Anthropic's, Cloudflare's, or any CDN's own infrastructure. No source-control history was available, so the review reflects the current state of the file. The

application is an educational planning tool and not financial, tax, or legal advice; it processes only the figures a user chooses to enter.

Prepared for internal security review of the AI Retirement Income Planner v7. Findings reflect the file state as of 4 June 2026.