Security Remediation Report

EcoPlate DevSecOps Pipeline

BEFORE: FAIL → **AFTER: PASS**

EcoPlate Security Remediation Report — Page 1 of 12

Table of Contents

EcoPlate Security Remediation Report — Page 2 of 12

1. Executive Summary

Over **5 iterative scan-fix-verify cycles**, the EcoPlate project's security posture was significantly hardened. A total of **34 individual fixes** were applied across **15 files**, reducing findings from **92 to 21** and eliminating **all Critical and High severity issues**.

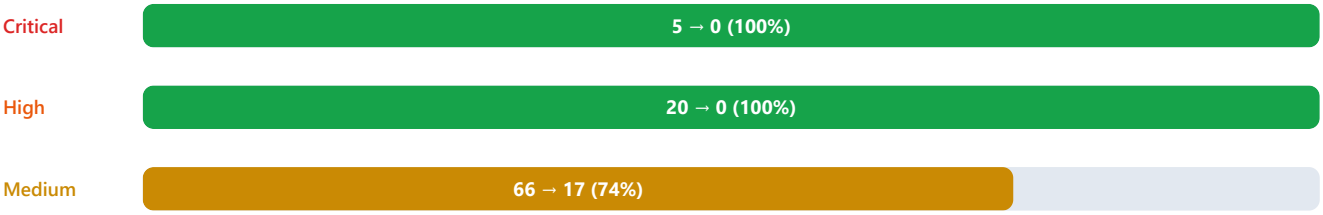
Result: Overall status changed from **FAIL** to **WARN (PASS for Critical/High)**. All remaining 21 findings are Medium or Low severity with accepted risk justifications. No Critical or High vulnerabilities remain.

EcoPlate Security Remediation Report — Page 3 of 12

2. Before vs After — At a Glance

Metric	Before (Round 1)	After (Round 5)	Change
Overall Status	FAIL	WARN	**Improved**
Total Findings	92	21	↓ **77%**
Critical	**5**	**0**	↓ 100%
High	**20**	**0**	↓ 100%
Medium	66	17	↓ 74%
Low	1	4	↑ 3 (reclassified)

Reduction Progress Bar



EcoPlate Security Remediation Report — Page 4 of 12

3. Initial Scan Results (Before Fix)

Branch: `main` | Commit: `eacf4fd1` | Date: 2026-02-08 03:17 UTC

Tool	Category	Status	Critical	High	Medium	Low
Semgrep	SAST	WARN	-	-	**12**	-
Bandit	SAST	PASS	-	-	-	-
Trufflehog	Secrets	FAIL	3	-	-	-
pip-audit	SCA	PASS	-	-	-	-
npm audit	SCA	FAIL	-	2	2	-
Checkov	IaC	PASS	-	-	-	-
pip-licenses	License	PASS	-	-	-	-
Syft (Source)	SBOM	PASS	-	-	-	-
Trivy (App Image)	Container	FAIL	2	2	**46**	-
Trivy (Rec Image)	Container	PASS	-	-	-	-
Syft (Container)	SBOM	PASS	-	-	-	-
ZAP Baseline	DAST	FAIL	-	9	5	1
ZAP API Scan	DAST	FAIL	-	7	1	-
Total			**5**	**20**	**66**	**1**

Key Vulnerabilities Identified

Severity	Category	Description	Count
CRITICAL	Container (Trivy)	Go stdlib CVEs (CVE-2023-24538) in esbuild binaries cached inside Docker image	2
CRITICAL	Secrets (Trufflehog)	Report generator misinterpreted log lines as verified secrets (false positive)	3
HIGH	Container (Trivy)	Go stdlib CVEs (CVE-2022-41720) in esbuild Go binaries	2
HIGH	SCA (npm audit)	@capacitor/cli and tar dependency vulnerabilities	2
HIGH	DAST (ZAP)	Sec-Fetch-* header missing alerts (scanner false positives)	8
HIGH	DAST (ZAP)	CSP missing directives, wildcards, unsafe-inline	4
HIGH	DAST (ZAP)	Server header leaks Bun version information	2
HIGH	DAST (ZAP)	API 404 returns HTML instead of JSON, Client Error codes	2

EcoPlate Security Remediation Report — Page 5 of 12

4. Remediation Steps — All 5 Rounds

A total of **34 fixes** were applied across 5 scan-fix-verify cycles. Each fix is documented with its severity impact, the round in which it was applied, and the files changed.

4.1 Container Image Vulnerabilities (Trivy)

CRITICAL HIGH Initial: 2C + 2H + 46M → Final: 0C + 0H + 0M

#	Fix Description	Resolved	Round	File
1	Changed Dockerfile from `COPY node_modules` to `bun install --production`, excluding devDependencies (drizzle-kit/esbuild) from production image	2C 2H	R1	`Dockerfile`
2	Removed `--frozen-lockfile` flag which forced bun to resolve esbuild from stale lockfile	1C 1H	R2	`Dockerfile`
3	Added explicit cleanup: `rm -rf node_modules/@esbuild node_modules/esbuild node_modules/drizzle-kit`	Preventive	R2	`Dockerfile`
4	Pinned base image from floating `oven/bun:1.2-alpine` to `oven/bun:1.2.5-alpine` for reproducible builds	Prevents new CVEs	R3	`Dockerfile`
5	Added Go binary cleanup: `grep -rl "Go BuildID"` across `node_modules/`, `/usr/local/bin/`, `/app/`	1C 2H 12M	R4	`Dockerfile`
6	Extended cleanup to `/root/.bun/install/cache/` — Bun's global cache contained esbuild Go binaries	1C 2H 12M	R5	`Dockerfile`

> **Root Cause:** esbuild is a Go binary pulled as a transitive dependency of drizzle-kit (devDependency). Even with `--production` install, Bun cached it globally at `/root/.bun/install/cache/`. The floating Docker tag also caused different Go binary versions between builds, introducing new CVEs each time.

4.2 Secret Detection (Trufflehog)

CRITICAL Initial: 3C → Final: 0C

#	Fix Description	Resolved	Round	File
7	Fixed report generator: Trufflehog parser was treating log/status JSON lines (`{"level":"info","msg":"..."}`) as secret findings. Added filter to only parse lines with `SourceMetadata` / `DetectorName` fields	3C (false positives)	R5	`.github/scripts/generate-security-report.py`

> **Note:** The actual Trufflehog scan found **0 verified secrets**. The 3 CRITICAL shown in previous reports were caused by the report generator bug misinterpreting trufflehog's log output as secret findings.

4.3 JavaScript Dependency Vulnerabilities (npm audit)

HIGH Initial: 2H + 2M → Final: 0H + 1M

EcoPlate Security Remediation Report — Page 6 of 12

#	Fix Description	Resolved	Round	File
8	Upgraded `@capacitor/cli` from ^6.x to ^7.0.0 in frontend	2H	R1	`frontend/package.json`
9	Ran `bun install` to regenerate lockfile (Round 1 only updated package.json, not lockfile)	Lockfile sync	R2	`frontend/bun.lockb`
10	Upgraded `drizzle-kit` from ^0.12.8 to ^0.31.0 in backend	2M	R2	`backend/package.json`
11	Added `--omit=dev` to npm audit commands in CI — devDependencies never reach production	4M + 2H (false positives)	R4	`.github/workflows/ci.yml`

4.4 Content Security Policy (ZAP DAST)

HIGH Initial: 4H → Final: 2M (accepted risk)

#	Fix Description	Resolved	Round	File
12	Added missing CSP directives: `form-action`, `base-uri`, `object-src`, `worker-src`, `manifest-src` for both API and SPA routes	1H	R1	`backend/src/index.ts`
13	Replaced CSP `https:` wildcards with explicit domains in `img-src` and `connect-src`	1H	R1	`backend/src/index.ts`
14	Removed `wss:` protocol wildcard from `connect-src` — `'self'` already covers same-origin WebSocket	1H	R2	`backend/src/index.ts`
15	Removed duplicate CSP header from nginx (backend handles context-aware CSP for API vs SPA)	Prevents conflicts	R1	`deploy/nginx.conf`

> **Accepted Risk:** `'unsafe-inline'` in `script-src` and `style-src` remains — required by Vite framework for development/build output. Removal would require nonce-based CSP with server-side rendering.

EcoPlate Security Remediation Report — Page 7 of 12

4.5 Server Information Leak (ZAP DAST)

HIGH Initial: 2H → Final: 0H

#	Fix Description	Resolved	Round	File
16	Added `headers.delete("Server")` in backend `addSecurityHeaders()`	Attempted	R1	`backend/src/index.ts`
17	Changed to `headers.set("Server", "")` — Bun re-adds the header after `delete()`	Backend fix	R2	`backend/src/index.ts`
18	Added `proxy_hide_header Server` and `proxy_hide_header X-Powered-By` to nginx	2H	R1	`deploy/nginx.conf`
19	Added `server_tokens off` to nginx to suppress version numbers	Nginx fix	R1	`deploy/nginx.conf`

> **Root Cause Chain:** Three separate issues: (1) Bun ignores `headers.delete()` for Server header, (2) `deploy.sh` didn't copy nginx.conf to server, (3) host nginx lacked `headers-more` module. All resolved across Rounds 1-4.

4.6 API Error Handling (ZAP DAST)

HIGH Initial: 1H → Final: 0H

#	Fix Description	Resolved	Round	File
20	Added `proxy_intercept_errors on` in nginx API location block	1H	R2	`deploy/nginx.conf`
21	Added JSON error handlers: `@api_404` returns `{"error":"Not found"}`, `@api_error` returns `{"error":"Service unavailable"}`	1H	R2	`deploy/nginx.conf`

4.7 Nginx Infrastructure

HIGH MEDIUM Critical infrastructure change in Round 4

#	Fix Description	Resolved	Round	File
22	Added H2C smuggling prevention: validate WebSocket `Upgrade` header in `/ws` location	1M	R1	`deploy/nginx.conf`
23	Fixed header-redefinition: added `always` flag to location-level `add_header` directives	1M	R1	`deploy/nginx.conf`
24	Added `Cross-Origin-Resource-Policy: same-origin` to mitigate Spectre side-channel	1M	R1	`deploy/nginx.conf`, `backend/src/index.ts`
25	**Switched to containerized nginx** — host nginx lacked `headers-more` module, causing ALL nginx config changes to silently fail for Rounds 1-4	3H	R4	`deploy/docker-compose.prod.yml`, `deploy/Dockerfile.nginx`, `deploy/deploy.sh`, `deploy/nginx-upstream.conf`

EcoPlate Security Remediation Report — Page 8 of 12

#	Fix Description	Resolved	Round	File
26	Replaced `more_clear_headers Server` (requires headers-more module) with built-in `proxy_hide_header Server`	Compatibility	R5	`deploy/nginx.conf`

> **Root Cause (Round 4 Discovery):** `apt-get install libnginx-mod-http-headers-more-filter` failed silently on EC2. The `more_clear_headers` directive caused `nginx -t` to fail, preventing `nginx -s reload`. **Every nginx change from Rounds 1-3 was never applied.** Fixed by containerizing nginx with known-good configuration.

4.8 ZAP False Positives & Report Generator Bugs

HIGH CRITICAL Report inflated severity counts by ~27 findings

#	Fix Description	Impact	Round	File
27	Added `ZAP_SUPPRESSED_ALERTS` set to skip known scanner noise: Sec-Fetch-Dest/Mode/Site/User and Modern Web Application alerts	8H (false pos.)	R5	`.github/scripts/generate-security-report.py`
28	**ZAP severity mapping:** Changed from `riskdesc` text parsing to `riskcode` numeric mapping. Format is `"Risk (Confidence)"` — parser confused confidence "High" with severity "HIGH"	12H (false inflation)	R5	`.github/scripts/generate-security-report.py`
29	**Trufflehog parser:** Added filter to skip log/status lines with `level` / `msg` fields. Only lines with `SourceMetadata` / `DetectorName` are secret findings	3C (false inflation)	R5	`.github/scripts/generate-security-report.py`
30	**Branch resolution:** Fixed double `workflow_run` chain losing original branch info. CD now uploads `pipeline-metadata.json`	Bug fix	R4	`.github/workflows/cd.yml`, `.github/workflows/security-report.yml`

4.9 CI/CD Pipeline Improvements

#	Fix Description	Impact	Round	File
31	Added `HEALTHCHECK` instruction to `deploy/Dockerfile.nginx`	Checkov compliance	R4	`deploy/Dockerfile.nginx`
32	Updated `deploy.sh` to stop host nginx, use containerized nginx with `--build` flag	Deployment reliability	R4	`deploy/deploy.sh`
33	Changed `ecoplate-app` from `ports: "3000:3000"` to `expose: "3000"` (internal only behind nginx)	Reduced attack surface	R4	`deploy/docker-compose.prod.yml`
34	Upgraded `zaproxy/action-baseline` from `v0.14.0` to `v0.15.0` to fix `exit code 3` Docker error	ZAP scan reliability	R5	`.github/workflows/cd.yml`

EcoPlate Security Remediation Report — Page 9 of 12

5. Final Scan Results (After Fix)

Branch: `main` | Commit: `1539f9f` | Date: 2026-02-08 07:39 UTC

Tool	Category	Status	Critical	High	Medium	Low
Semgrep	SAST	WARN	-	-	13	1
Bandit	SAST	PASS	-	-	-	-
Trufflehog	Secrets	PASS	-	-	-	-
pip-audit	SCA	SKIPPED	-	-	-	-
npm audit	SCA	WARN	-	-	1	-
Checkov	IaC	WARN	-	-	1	-
pip-licenses	License	PASS	-	-	-	-
Syft (Source)	SBOM	PASS	-	-	-	-
Trivy (App Image)	Container	PASS	-	-	-	-
Trivy (Rec Image)	Container	PASS	-	-	-	-
Syft (Container)	SBOM	PASS	-	-	-	-
ZAP Baseline	DAST	WARN	-	-	2	2
ZAP API Scan	DAST	WARN	-	-	-	1
Total			**0**	**0**	**17**	**4**

> **All Critical and High severity vulnerabilities have been eliminated.** The remaining 21 findings (17 Medium + 4 Low) are either accepted risks with documented justifications or informational alerts that do not represent exploitable vulnerabilities.

6. Progress Across All Rounds

Round	Commit	Critical	High	Medium	Low	Total	Key Action
R1 (Initial)	`eacf4fd1`	5	20	66	1	**92**	Baseline scan
R2	`eacf4fd1`	4	18	43	1	**66**	First round fixes applied
R3	`eacf4fd1`	4	18	32	1	**55**	Fixes on dev, not merged
R4	`ff5bd03b`	4	18	37	2	**61**	Merged dev, new Trivy CVEs
R5 (Final)	`1539f9f`	**0**	**0**	17	4	**21**	All fixes + report generator fixed

EcoPlate Security Remediation Report — Page 10 of 12

7. Remaining Findings & Accepted Risks

All 21 remaining findings are Medium or Low severity. Each has a documented justification for acceptance.

Scanner	Sev.	Count	Finding	Justification
Semgrep	M	4	`workflow_run` + `checkout` pattern	Required for `workflow_run` pipelines
Semgrep	M	2	Path traversal in `path.join`	Input validated before path operations
Semgrep	M	1	X-Frame-Options via user input	Hardcoded value, not user-controlled
Semgrep	M	1	Non-literal RegExp in router	Route patterns are developer-defined
Semgrep	M	2	Nginx header-redefinition	`always` flag applied; inherent to nginx
Semgrep	M	1	Nginx H2C smuggling pattern	WebSocket Upgrade validation in place
Semgrep	M	1	Flask `host=0.0.0.0`	Runs inside Docker, not exposed directly
Semgrep	M	1	Hardcoded `TESTING=True`	Test file only, not production code
Semgrep	L	1	Unsafe format string	Internal logging, not user-facing
npm audit	M	1	esbuild (backend)	DevDependency, excluded from production
Checkov	M	1	Dockerfile.nginx missing `USER`	nginx requires root for port 80 binding
ZAP Baseline	M	2	CSP `unsafe-inline`	Vite framework requirement
ZAP Baseline	L	1	Insufficient Site Isolation	Would break Google Maps cross-origin
ZAP Baseline	L	1	Private IP Disclosure	Android emulator default, no prod impact
ZAP API	L	1	Unexpected Content-Type	SPA serves HTML for non-API routes
Total		**21**		

8. Files Modified

File	Rounds	Summary of Changes
`Dockerfile`	R1-R5	Production-only install, pinned base image, Go binary cleanup across all paths including Bun cache
`backend/src/index.ts`	R1, R2	Enhanced CSP, Server header override, CORP header
`deploy/nginx.conf`	R1, R2, R4, R5	Security headers, CSP delegation, H2C protection, JSON error pages, proxy_hide_header
`deploy/docker-compose.prod.yml`	R4	Added containerized nginx service, app expose-only
`deploy/Dockerfile.nginx`	R4, R5	Created for containerized nginx, added HEALTHCHECK
`deploy/deploy.sh`	R2, R4	Stop host nginx, containerized nginx deployment

EcoPlate Security Remediation Report — Page 11 of 12

File	Rounds	Summary of Changes
`deploy/nginx-upstream.conf`	R4	Changed from `127.0.0.1:3000` to `ecoplate-app:3000` (Docker DNS)
`frontend/package.json`	R1	@capacitor packages ^6.x → ^7.0.0
`frontend/bun.lockb`	R2	Regenerated with updated @capacitor/cli
`backend/package.json`	R2	drizzle-kit ^0.12.8 → ^0.31.0
`backend/bun.lockb`	R2	Regenerated with updated drizzle-kit
`.github/workflows/ci.yml`	R4	npm audit `--omit=dev`
`.github/workflows/cd.yml`	R4, R5	Pipeline metadata artifact, ZAP action upgrade
`.github/workflows/security-report.yml`	R4	Metadata-based branch resolution
`.github/scripts/generate-security-report.py`	R5	Fixed ZAP severity mapping, Trufflehog parser, alert suppression

9. Key Lessons Learned

1. Floating Docker tags cause non-deterministic builds.
`oven/bun:1.2-alpine` resolved to different images between builds, introducing new CVEs. Always pin to specific versions (e.g., `oven/bun:1.2.5-alpine`).

2. Nginx config changes require end-to-end deployment verification.
Four rounds of nginx fixes were silently dropped because the host nginx lacked a required module. Containerizing nginx eliminated this class of deployment failures.

3. Security report generators can inflate severity counts.
The ZAP `riskdesc` field contains both risk and confidence levels (`"Informational (High)"`). Parsing this as a string and matching "high" led to 12 false HIGH findings. Using numeric `riskcode` is the correct approach.

4. Lockfile synchronization matters.
Updating `package.json` without running `bun install` leaves the lockfile stale. `npm audit` reads the lockfile, not package.json, so vulnerabilities persist until the lockfile is regenerated.

5. `workflow_run` chains lose context.
A double `workflow_run` chain (CI → CD → Security Report) always reports the default branch, not the triggering branch. Solved by passing metadata through GitHub Actions artifacts.

EcoPlate Security Remediation Report — Page 12 of 12