

VIBE96 QUALITY REPORT

EventTag Badge Platform

Event badge printing platform with admin portals, badge designer and Windows self-service kiosks.

Product ID	**B0004-15376**
Builder	**CodeHarbor**
Version reviewed	**1.0.0 · released 12 Nov 2026**
Stack	**ASP.NET Core MVC on .NET 10 (Razor views) + WPF kiosk app · C# / .NET 10 · SQL Server 2019+ (ADO.NET, parameterized SQL, no ORM)**
Built with	**Claude Code**
Review period	**12 Nov 2026 – 12 Nov 2026**

74 / 100

Listed after a manual review

Pass mark 75 and no failing check.

Scores

Security	77
Code structure	84
Docs & completeness	60
Dependency health	72
Build & deploy	62

✓ **AI code audit**
6 checks · 11 Nov 2026

✓ **Installed & run**
12 Nov 2026

✓ **Demo vs delivered**
12 Nov 2026

✓ **Final sign-off**
12 Nov 2026

At a glance

- ✓ Installed and run from the delivered code — 8 of 8 checks answered yes.
- ✓ Live demo compared with the delivered code — 7 of 7 checks answered yes.
- ! 9 of 15 promised items delivered (setup guide, example config and the rest).
- ✓ Security scored 77/100 with no failing check.
- ! 6 open warnings — explained in part 1.

Check this report is genuine. Scan the code or open vibe96.com/report/QR-26-00128. The online copy always shows the latest status.

PART 1

AI code audit

The full source archive was read by our AI auditor against each check. It reads the code — it does not run it; running it is part 2.

CHECK	WHAT WE FOUND	SCORE	RESULT
Secrets & credentials	Every regex candidate is a placeholder, an environment variable reference or application code handling tokens and passwords, and no live key, private key or real connection string was found. Docker Compose requires the passwords to be set, and the committed connection strings use Windows authentication. The remaining issues are weak defaults that ship to buyers: a documented demo password, a documented kiosk staff PIN, and a dev-certificate flag that is enabled by default.	88	Warning
Dependency vulnerabilities	The project uses central package management (Directory.Packages.props) and the README states no vulnerable packages are reported, but no packages.lock.json appears in the file tree, so restores are not reproducible. The actual package versions were not in the digest, so no specific CVE could be confirmed or ruled out.	72	Warning
Authorisation & access control	The documented architecture has per-portal cookie auth, a guard filter (sign-in, event assignment, [RequirePermission]), a hashed kiosk bearer token and BCrypt password hashes. The controller bodies were not in the digest, so per-action guards and id-based ownership checks (event scoping) could not be traced. Several secondary risks are visible: a default kiosk staff PIN, a [hidden] unsalted PIN hash, and an unauthenticated-by-design kiosk API that must be verified	70	Warning
Database access rules	The digest shows a single ADO.NET entry point (Db.cs) with parameterized SQL claims and an idempotent upgrade script, which is a good foundation. The repository and schema SQL bodies were not included in the excerpts, so injection safety, WHERE scoping on deletes/updates, per-event tenant scoping and FK indexes could not be confirmed. Several structural issues, such as documentation/file mismatches and a destructive schema script, lower confidence.	72	Warning
Code structure & duplication	The solution is split into sensible projects (Domain, Infrastructure, Admin, Enabler, Kiosk.Core, Web.Shared) with a documented architecture and a separate Kiosk.Core for testable logic. The main weaknesses are very large single-file browser scripts (designer.js, kiosk-admin.js) and a shipped design-reference folder with duplicated HTML prototypes that adds bulk for buyers.	84	Warning
Build reproducibility	The solution layout, central package management and Docker setup look coherent, and the demo reset path exists. However, several referenced items (schema.sql, seed.sql, a project file, the upgrade command) are not in the visible tree, and the docs disagree on database script names, so a clean-machine setup may break at the database step. HowToSetup.md and the Dockerfile were not shown, so setup order and SDK pinning could not be fully verified.	62	Warning

Open warnings

Problems that did not stop the listing. They stay on the product page until a later version fixes them.

Warning 1 **No real credentials found; demo password, demo PIN and permissive dev defaults ship to buyers**

Every regex candidate is a placeholder, an environment variable reference or application code handling tokens and passwords, and no live key, private key or real connection string was found. Docker Compose requires the passwords to be set, and the committed connection strings use Windows authentication. The remaining issues are weak defaults that ship to buyers: a documented demo password, a documented kiosk staff PIN, and a dev-certificate flag that is enabled by default.

Warning 2 **NuGet versions are centrally pinned, but no lockfile and package contents were not visible**

The project uses central package management (Directory.Packages.props) and the README states no vulnerable packages are reported, but no packages.lock.json appears in the file tree, so restores are not reproducible. The actual package versions were not in the digest, so no specific CVE could be confirmed or ruled out.

Warning 3 **Authz design looks sound but controllers were not provided, so guards and ownership checks are unverified**

The documented architecture has per-portal cookie auth, a guard filter (sign-in, event assignment, [RequirePermission]), a hashed kiosk bearer token and BCrypt password hashes. The controller bodies were not in the digest, so per-action guards and id-based ownership checks (event scoping) could not be traced. Several secondary risks are visible: a default kiosk staff PIN, a [hidden] unsalted PIN hash, and an unauthenticated-by-design kiosk API that must be verified

Warning 4 **Well-modularised solution with clear portal and layer separation; some oversized JS files and hard-coded demo values**

The solution is split into sensible projects (Domain, Infrastructure, Admin, Enabler, Kiosk.Core, Web.Shared) with a documented architecture and a separate Kiosk.Core for testable logic. The main weaknesses are very large single-file browser scripts (designer.js, kiosk-admin.js) and a shipped design-reference folder with duplicated HTML prototypes that adds bulk for buyers.

Warning 5 **Build likely works, but docs and scripts reference files and tools not visible or inconsistent with the tree**

The solution layout, central package management and Docker setup look coherent, and the demo reset path exists. However, several referenced items (schema.sql, seed.sql, a project file, the upgrade command) are not in the visible tree, and the docs disagree on database script names, so a clean-machine setup may break at the database step. HowToSetup.md and the Dockerfile were not shown, so setup order and SDK pinning could not be fully verified.

Warning 6 **Parameterized ADO.NET evident, but SQL source and schema were mostly not visible; some gaps and unverified paths**

The digest shows a single ADO.NET entry point (Db.cs) with parameterized SQL claims and an idempotent upgrade script, which is a good foundation. The repository and schema SQL bodies were not included in the excerpts, so injection safety, WHERE scoping on deletes/updates, per-event tenant scoping and FK indexes could not be confirmed. Several structural issues, such as documentation/file mismatches and a destructive schema script, lower confidence.

Requirement list

What the builder agreed to deliver before uploading — 9 of 15 met.

!	Complete source for every component	Partial
!	No real credentials in the archive	Partial
!	Builds from a clean machine	Partial
✓	Database script or migrations	Met
!	Authentication and roles on every mutating endpoint	Partial
!	Pinned dependencies without critical CVEs (nice)	Partial
✓	Documentation set (nice)	Met
!	Working live demo	Partial
✓	README.md	Met
✓	HowToSetup.md	Met

✓	CLAUDE.md (or AGENTS.md) (nice)	Met
✓	CHANGELOG.md (nice)	Met
✓	appsettings.example.json	Met
✓	a project file	Met
✓	a project file (nice)	Met

PART 2

Human review

Reviewers from the Vibe96 team in Dubai worked through each question below; a final reviewer signed it off.

Stage 1 — Installed and run from the delivered code

Completed 12 Nov 2026

A reviewer installed the product on a clean machine, following only the guide that ships with it.

QUESTION	ANSWER	REVIEWER'S NOTE
Packages restored without errors	Yes	
Solution builds clean	Yes	
Database created from the shipped script or migrations	Yes	
First sign-in works with the documented steps	Yes	
Setup guide is accurate and complete	Yes	
Every module in the feature list opens and works	Yes	
No real secrets in the archive	Yes	
Branding and company details can be changed without code	Yes	

Stage 2 — Live demo compared with the delivered code

Completed 12 Nov 2026

A second reviewer checked that what the listing and demo show is what the buyer receives.

QUESTION	ANSWER	REVIEWER'S NOTE
Demo URL is reachable	Yes	
Demo credentials work	Yes	
Demo runs the same version as the archive	Yes	
All five differentiators are visible in the demo	Yes	
Screenshots match the demo	Yes	
Nothing in the demo is missing from the code	Yes	
Datasheet claims are accurate	Yes	

Changes asked for during review

None — approved on the first round.

Sign-off: approved for sale on 12 Nov 2026 with 6 warnings shown publicly.

What this report covers — and what it doesn't

- **It covers** version 1.0.0 exactly as the buyer downloads it: the source archive, the setup guide and the live demo on the dates above.

- **It is not** a penetration test, a performance or load test, or a legal review of third-party package licences.
- **Scores are a point in time.** Libraries age; a later version is audited again and gets its own report.
- **Your setup matters.** Hosting, configuration and changes made after purchase are outside this review.
- **How the score is made:** each AI check is scored 0–100 and weighted (security counts most). A listing needs 75 or more, no failing check, and the human review passed.
- **Always try the live demo** before buying, and ask the builder questions from the product page.