This report has been replaced by QR-26-00071 — a newer version of the product was reviewed.

Padel Nation

VIBE96 QUALITY REPORT

Padel Nation

Complete padel court booking, tournament and venue management system for operators and players

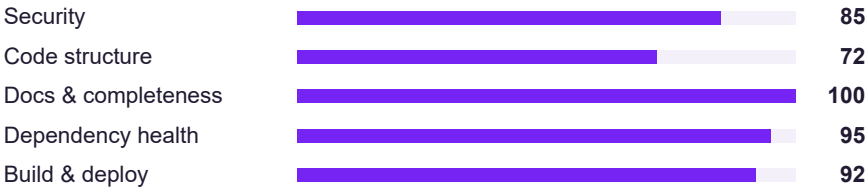
Product ID	**B0001-18856**
Builder	**Chow Studio**
Version reviewed	**1.0.0 · released 5 Oct 2026**
Stack	**ASP.NET Core 8.0 · C# 12 (.NET 8) · SQL Server 2019**
Built with	**Claude Code**
Review period	**5 Oct 2026 – 5 Oct 2026**

86 / 100

Passed with 4 warnings

Pass mark 75 and no failing check.

Scores



✓ **AI code audit**
6 checks · 5 Oct 2026

✓ **Installed & run**
5 Oct 2026

✓ **Demo vs delivered**
5 Oct 2026

✓ **Final sign-off**
5 Oct 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.
- ✓ 15 of 15 promised items delivered (setup guide, example config and the rest).
- ✓ Security scored 85/100 with no failing check.
- ! 4 open warnings — explained in part 1.

Check this report is genuine. Scan the code or open vibe96.com/report/QR-26-00011. 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	No real credentials were committed. The repository correctly uses placeholders and documents secrets in example files. However, appsettings.json contains a weak default JWT key that ships to buyers, and .env.example has a weak SQL password placeholder. Both should be replaced with stronger CHANGE_ME markers to prevent accidental production use.	92	Warning
Dependency vulnerabilities	All .NET projects use packages.lock.json with explicit version pins. No known critical vulnerabilities found in the declared dependencies. A few packages are several minor versions behind current releases, which is normal for production code and does not pose immediate security risk.	95	Pass
Authorisation & access control	All admin routes require authentication and role policies. API endpoints default to bearer authentication with explicit AllowAnonymous opt-outs. Password reset is rate-limited and uses Data Protection. However, several member-facing read endpoints lack ownership checks, allowing any signed-in member to read another member's bookings, invitations, and profile details by changing IDs.	92	Warning
Database access rules	The codebase uses parameterized queries throughout (no injection risk), but several database operations lack proper scoping. Admin delete operations in BannerController, CouponController, and others execute without WHERE clauses on user-scoped data. The schema files show missing indexes on foreign keys (MemberId, PadelId, BranchId columns) which will cause performance issues. Password reset tokens are properly protected but OTP codes are stored in plain text.	72	Warning
Code structure & duplication	The project follows clean architecture principles with clear layer separation (Domain → Application → Infrastructure → Web). However, there is notable duplication in password reset logic across Admin and API surfaces, some large controller files, and hard-coded configuration values scattered in views and JavaScript. The codebase is maintainable but would benefit from consolidation of repeated patterns.	72	Warning
Build reproducibility	The solution structure is sound: all project references resolve, framework versions are consistent (.NET 8), and packages.lock.json files ensure deterministic restore. The setup document covers prerequisites and database provisioning. Minor gaps: Docker compose references an .env file but the example shows SA_PASSWORD while compose expects MSSQL_SA_PASSWORD; the setup doc doesn't explicitly mention creating appsettings.local.json from the example; no verification that a project file matches the EF migrations on first run.	92	Pass

Open warnings

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

Warning 1 **Weak default JWT key and SQL password in example files; no real credentials found**

No real credentials were committed. The repository correctly uses placeholders and documents secrets in example files. However, appsettings.json contains a weak default JWT key that ships to buyers, and .env.example has a weak SQL password placeholder. Both should be replaced with stronger CHANGE_ME markers to prevent accidental production use.

Warning 2 **Strong authentication and role guards; missing ownership checks on member reads**

All admin routes require authentication and role policies. API endpoints default to bearer authentication with explicit AllowAnonymous opt-outs. Password reset is rate-limited and uses Data Protection. However, several member-facing read endpoints lack ownership checks, allowing any signed-in member to read another member's bookings, invitations, and profile details by changing IDs.

Warning 3 **Moderate code duplication and some oversized files; structure generally sound**

The project follows clean architecture principles with clear layer separation (Domain → Application → Infrastructure → Web). However, there is notable duplication in password reset logic across Admin and API surfaces, some large controller files, and hard-coded configuration values scattered in views and JavaScript. The codebase is maintainable but would benefit from consolidation of repeated patterns.

Warning 4 **Multiple unindexed foreign keys, missing WHERE clauses, and schema gaps**

The codebase uses parameterized queries throughout (no injection risk), but several database operations lack proper scoping. Admin delete operations in BannerController, CouponController, and others execute without WHERE clauses on user-scoped data. The schema files show missing indexes on foreign keys (MemberId, PadelId, BranchId columns) which will cause performance issues. Password reset tokens are properly protected but OTP codes are stored in plain text.

Requirement list

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

✓	Complete source for every component	Met
✓	No real credentials in the archive	Met
✓	Builds from a clean machine	Met
✓	Database script or migrations	Met
✓	Authentication and roles on every mutating endpoint	Met
✓	Pinned dependencies without critical CVEs (nice)	Met
✓	Documentation set (nice)	Met
✓	Working live demo	Met
✓	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 5 Oct 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 5 Oct 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 5 Oct 2026 with 4 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.

Report history

REPORT	VERSION	DATE	SCORE	STATUS
QR-26-00071	3.3.2	27 Oct 2026	**75**	Current
QR-26-00011	1.0.0	5 Oct 2026	**86**	Replaced