

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	**3.3.2 · released 27 Oct 2026**
Stack	**ASP.NET Core 8.0 · C# 12 (.NET 8) · SQL Server 2019**
Built with	**Claude Code**
Review period	**27 Oct 2026 – 27 Oct 2026**

75 / 100

Passed with 6 warnings

Pass mark 75 and no failing check.

Scores

Security	74
Code structure	72
Docs & completeness	60
Dependency health	78
Build & deploy	80

✓ **AI code audit**
6 checks · 27 Oct 2026

✓ **Installed & run**
27 Oct 2026

✓ **Demo vs delivered**
27 Oct 2026

✓ **Final sign-off**
27 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.

! 9 of 15 promised items delivered (setup guide, example config and the rest).

! Security scored 74/100 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-26-00071. 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 env-var reference, or code that handles secrets without containing one. The only real concern is a well-known development JWT key in appsettings.Development.json that is published to every buyer. Production config refuses to start with the CHANGE_ME key, and integration secrets are encrypted and stored via admin settings.	**90**	Warning
Dependency vulnerabilities	Each .NET project (Domain, Application, Infrastructure, Web) ships a packages.lock.json, and a DEPENDENCY-AUDIT.md exists, which is a good sign. The digest did not include the .csproj or lockfile contents, so no package version could be checked against known advisories. The score is capped until the versions can be reviewed, and it should not be read as a clean bill of health.	**78**	Warning
Authorisation & access control	The visible evidence points to sound authz: passwords go through a hasher, the API base class uses a Member policy, admin controllers are documented as [Authorize], and enumeration is masked. Only the AuthService code and docs were provided, not the controllers, so per-endpoint guards and ownership checks (e.g. bookings by RowKey, payment callbacks, Diagnostics and Setting) could not be traced. Several risk areas remain unverified.	**62**	Warning
Database access rules	The visible AuthService uses parameterised EF Core LINQ, and docs state Dapper reads are parameterised. However, the Dapper report SQL, backup/ restore code, DemoDataResetService (deletes every row) and schema scripts were not in the digest, so injection, unscoped deletes, FKs and indexes could not be verified. Plaintext-looking secret columns and a few query patterns are flagged as warnings.	**70**	Warning
Code structure & duplication	The solution separates Domain, Application, Infrastructure and Web as documented, which is a sound structure. However, the digest shows very large seed/demo SQL files and service classes with many concerns, and configuration defaults are scattered across several files.	**72**	Warning
Build reproducibility	The solution layout is coherent: four projects with lockfiles, global.json, Directory.Build.props, a Dockerfile and compose with a required SA password. Setup is documented with database scripts and configuration steps, but the base appsettings deliberately refuses to start until Jwt:Key is set, and a few project-level details could not be verified from the digest.	**80**	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; one published dev JWT key and a weak 'sa' compose default noted**

Every regex candidate is a placeholder, an env-var reference, or code that handles secrets without containing one. The only real concern is a well-known development JWT key in appsettings.Development.json that is published to every buyer. Production config refuses to start with the CHANGE_ME key, and integration secrets are encrypted and stored via admin settings.

Warning 2 **Layered solution with clear projects, but large files and secrets-adjacent config sprawl hurt maintainability**

The solution separates Domain, Application, Infrastructure and Web as documented, which is a sound structure. However, the digest shows very large seed/demo SQL files and service classes with many concerns, and configuration defaults are scattered across several files.

Warning 3 **Build likely reproducible; digest lacks csproj/Dockerfile/HowToSetup contents, a few gaps remain**

The solution layout is coherent: four projects with lockfiles, global.json, Directory.Build.props, a Dockerfile and compose with a required SA password. Setup is documented with database scripts and configuration steps, but the base appsettings deliberately refuses to start until Jwt:Key is set, and a few project-level details could not be verified from the digest.

Warning 4 **Lockfiles present for all four projects, but contents were not visible so versions could not be verified**

Each .NET project (Domain, Application, Infrastructure, Web) ships a packages.lock.json, and a DEPENDENCY-AUDIT.md exists, which is a good sign. The digest did not include the .csproj or lockfile contents, so no package version could be checked against known advisories. The score is capped until the versions can be reviewed, and it should not be read as a clean bill of health.

Warning 5 **No injection seen in the visible excerpts, but most data-access code and the schema scripts were not reviewable**

The visible AuthService uses parameterised EF Core LINQ, and docs state Dapper reads are parameterised. However, the Dapper report SQL, backup/restore code, DemoDataResetService (deletes every row) and schema scripts were not in the digest, so injection, unscoped deletes, FKs and indexes could not be verified. Plaintext-looking secret columns and a few query patterns are flagged as warnings.

Warning 6 **Authz design looks sound, but most controllers were not provided, so coverage cannot be verified**

The visible evidence points to sound authz: passwords go through a hasher, the API base class uses a Member policy, admin controllers are documented as [Authorize], and enumeration is masked. Only the AuthService code and docs were provided, not the controllers, so per-endpoint guards and ownership checks (e.g. bookings by RowKey, payment callbacks, Diagnostics and Setting) could not be traced. Several risk areas remain unverified.

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 27 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 27 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 27 Oct 2026 with 6 warnings shown publicly.

What this report covers — and what it doesn't

- **It covers** version 3.3.2 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