This report has been replaced by QR-26-00050 — a newer version of the product was reviewed.



VIBE96 QUALITY REPORT

WashBay

Car wash center management with customer booking, staff operations, payments and web admin for multi-branch businesses

Product ID	**B0003-91197**
Builder	**Max code**
Version reviewed	**1.0.0 · released 19 Oct 2026**
Stack	**ASP.NET Core 10 + Flutter 3.35 · C# 13 / .NET 10 · Microsoft SQL Server 2019+**
Built with	**Claude Code**
Review period	**16 Oct 2026 – 19 Oct 2026**

87 / 100

Passed with 6 warnings

Pass mark 75 and no failing check.

Scores

Security	92
Code structure	72
Docs & completeness	88
Dependency health	85
Build & deploy	78

✓ **AI code audit**
6 checks · 15 Oct 2026

✓ **Installed & run**
19 Oct 2026

✓ **Demo vs delivered**
19 Oct 2026

✓ **Final sign-off**
19 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 17 promised items delivered (setup guide, example config and the rest).
- ✓ Security scored 92/100 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-26-00040. 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 API keys or secrets were committed. However, HowToSetup.md and .env.example contain weak default passwords (e.g. '[hidden]', 'Manager@12345') that could be copied verbatim by users. The appsettings.Development.json ships with demo passwords enabled by default. All are documented as examples, but stronger placeholders would prevent accidental production use.	92	Warning
Dependency vulnerabilities	All lockfiles are present and no known critical vulnerabilities were found. The Flutter app uses slightly outdated packages (flutter_bloc 8.1.6, dio 5.7.0) that have newer versions available, and the .NET project uses ASP.NET Core 10 (pre-release) which should be updated to a stable release. These are maintenance issues, not security emergencies.	85	Warning
Authorisation & access control	All admin, staff and mutating endpoints require authentication and role checks via ASP.NET Core policies. Password hashing uses BCrypt. The main gaps are missing ownership checks on a few customer read endpoints (vehicles, bookings, notifications) where user A could read user B's data by changing an ID in the URL. These are warnings because they leak only customer data, not admin data or payment details.	92	Warning
Database access rules	No SQL injection or unscoped destructive statements found. EF Core migrations are complete and use parameterized queries throughout. The schema has proper foreign keys and cascade rules. Minor issue: several foreign key columns lack explicit indexes (EF creates them on navigation properties but not all FK columns have them), which may affect join performance at scale.	92	Warning
Code structure & duplication	The project has clear module boundaries and good separation of concerns overall. However, there is noticeable duplication in Flutter network/auth code, some large files mixing concerns, and hard-coded values scattered across configuration. The .NET side is well-structured; the Flutter side has room for consolidation.	72	Warning
Build reproducibility	The project can build on a clean machine but HowToSetup.md has critical gaps. The Flutter app requires env.json creation (mentioned in Flutter a project file but not in main setup), SQL Server setup jumps from Docker run to dotnet ef without creating the washbay user, and the .NET restore step is implied but never explicit. Docker Compose references .env variables that must be set before first run.	78	Warning

Open warnings

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

Warning 1 **Database access is safe; minor schema gaps (missing indexes on some foreign keys)**

No SQL injection or unscoped destructive statements found. EF Core migrations are complete and use parameterized queries throughout. The schema has proper foreign keys and cascade rules. Minor issue: several foreign key columns lack explicit indexes (EF creates them on navigation properties but not all FK columns have them), which may affect join performance at scale.

Warning 2 **No critical vulnerabilities; Flutter and .NET dependencies are outdated but not exploitable**

All lockfiles are present and no known critical vulnerabilities were found. The Flutter app uses slightly outdated packages (flutter_bloc 8.1.6, dio 5.7.0) that have newer versions available, and the .NET project uses ASP.NET Core 10 (pre-release) which should be updated to a stable release. These are maintenance issues, not security emergencies.

Warning 3 **Weak default passwords in example files; no real credentials found**

No real API keys or secrets were committed. However, HowToSetup.md and .env.example contain weak default passwords (e.g. '[hidden]', 'Manager@12345') that could be copied verbatim by users. The appsettings.Development.json ships with demo passwords enabled by default. All are documented as examples, but stronger placeholders would prevent accidental production use.

Warning 4 **Strong auth guards on mutating endpoints; minor ownership gaps on read-only customer data**

All admin, staff and mutating endpoints require authentication and role checks via ASP.NET Core policies. Password hashing uses BCrypt. The main gaps are missing ownership checks on a few customer read endpoints (vehicles, bookings, notifications) where user A could read user B's data by changing an ID in the URL. These are warnings because they leak only customer data, not admin data or payment details.

Warning 5 **Moderate duplication and some oversized files; structure is sound but maintainability could improve**

The project has clear module boundaries and good separation of concerns overall. However, there is noticeable duplication in Flutter network/auth code, some large files mixing concerns, and hard-coded values scattered across configuration. The .NET side is well-structured; the Flutter side has room for consolidation.

Warning 6 **Build reproducible with gaps: missing Flutter config step, unclear SQL setup, no .NET restore verification**

The project can build on a clean machine but HowToSetup.md has critical gaps. The Flutter app requires env.json creation (mentioned in Flutter a project file but not in main setup), SQL Server setup jumps from Docker run to dotnet ef without creating the washbay user, and the .NET restore step is implied but never explicit. Docker Compose references .env variables that must be set before first run.

Requirement list

What the builder agreed to deliver before uploading — 15 of 17 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	Partial
!	Pinned dependencies without critical CVEs (nice)	Partial
✓	Documentation set (nice)	Met
✓	Working live demo	Met
✓	Mobile app buildable and pointed at a configurable API	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
✓	a project file	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 19 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 19 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 19 Oct 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.

Report history

REPORT	VERSION	DATE	SCORE	STATUS
QR-26-00050	1.1.0	22 Oct 2026	**89**	Current
QR-26-00040	1.0.0	19 Oct 2026	**87**	Replaced