

VisualBase POC Technical Evaluation

Inventory and production platform decision before development begins

Decision	Current recommendation
Platform choice	Use .NET 10 LTS and ABP as the foundation for a new NCGR-owned product.
VisualBase	Do not start strategic inventory or production development on the current POC evidence.
Reconsideration	Keep VisualBase conditional. The vendor would need to pass the technical and source-handover tests in this report.

Why this decision is needed now. Inventory and production product implementation has not started. The POC was requested to choose the foundation before NCGR commits product rules, data structures, and operating processes to one platform. A fast demonstration of forms is useful evidence, but it does not settle whether the future system will keep stock and production records correct under stress. [S4][S5][S11]

Decision status. This is an engineering recommendation, not a ratified NCGR architecture decision or a production-readiness certification. Missing evidence means a capability is unproven, not impossible. The source-code promise and deployment topology are stakeholder-reported; the agreement, legal rights, and full service inventory were not reviewed. [S5][S12]

1 The answer in plain language

VisualBase showed that a team can create database-backed forms and configure approval steps quickly. In the recording, a table definition and SQL preview, field formula, generated form, approval settings, and a SQL stored procedure were handled inside one development workflow. That is a real advantage for routine data applications. [S4][S11]

Inventory and production have a harder job. Two users may try to reserve the same quantity. An integration may retry a receipt after a timeout. An order may use an older approved bill of materials. The system must give the correct business result every time and leave an audit trail. This review found no evidence that the POC tested those cases. [S4][S11]

The architecture adds a second concern. VisualBase behavior can be defined across platform metadata (configuration records that tell the engine what to build), validation tables, SQL views and procedures, and the runtime engine. A team needs a reliable way to review and test one complete change across those places. The POC did not demonstrate that release path. This is a governance gap in the evidence, not a claim that SQL Server or low-code is inherently unsuitable. [S1][S2][S4][S11]

The stakeholder working with the VisualBase team on deployment in NCGR infrastructure reports **more than 20 services**, not one backend and one frontend. Several services reportedly have their own user interface; render.visualbase.com and workflow.visualbase.com were given as examples. This broadens the handover question: NCGR needs an agreed model for operating and troubleshooting the connected estate, including failures that cross service boundaries. The exact inventory and topology still require independent verification. [S12]

Plain-language finding	Meaning for the decision
Fast to make screens	A genuine benefit for forms, approval steps, and routine administration; it does not prove stock or production correctness.
Rules are spread out	A change may involve configuration records, SQL objects, and platform behavior. NCGR needs one way to review and test the whole change.
Many reported services	The platform cannot be treated as one website for handover. NCGR needs a verified component map and an independent support test.

2 What the POC showed

The assessment uses four evidence labels. **Observed** means visible in the NCGR POC or recording. **Vendor-described** means stated in public Visualsoft material. **Stakeholder-reported** covers the expected source handover and the deployment work. **Not demonstrated** means this review did not see enough proof; it does not mean the feature is absent. [S1][S4][S5][S11][S12]

Evidence	What was observed	What it does not establish
NCGR demo [S4]	Business modules, inventory and production menu items, generated grids, object configuration, security and approval options.	Correct transaction behavior or enforcement of every configuration option.
Recording ~40:00 [S11]	Quick Table previewed SQL CREATE TABLE statements, including audit columns.	Execution of that DDL, versioned migration, peer review, or environment promotion.
Recording ~50:00 [S11]	Formula Editor offered local Test and Validate actions.	Regression tests for all rules, SQL, workflows, and integrations.
Recording ~72:30 [S11]	SQL Studio showed direct editing of an approval stored procedure.	A reviewed and repeatable release process or a production defect.

What the recording proves visually

Quick Table showed a table definition and SQL DDL preview in the session. The Formula Editor offered local Test and Validate actions while a field formula was being written. Approval settings and a generated form were also visible. These are strong signs of development speed, not evidence that the resulting business process is safe under every failure condition. [S11]

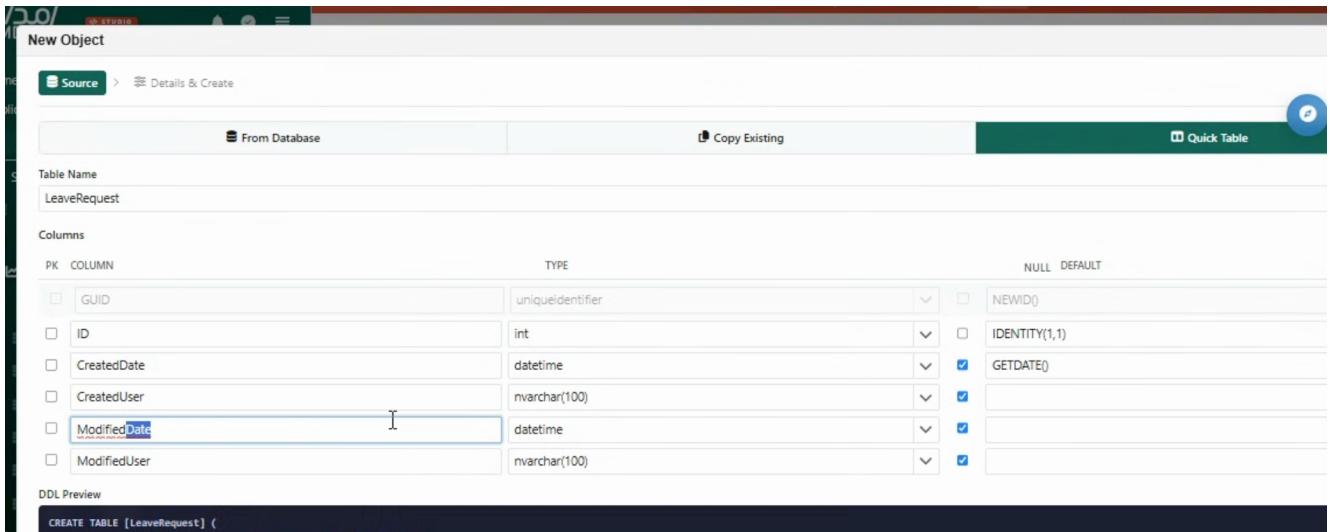


Figure 1. Quick Table shows a table definition and SQL DDL preview, about 40 minutes into the recording. This frame does not prove the DDL was executed. [S11]

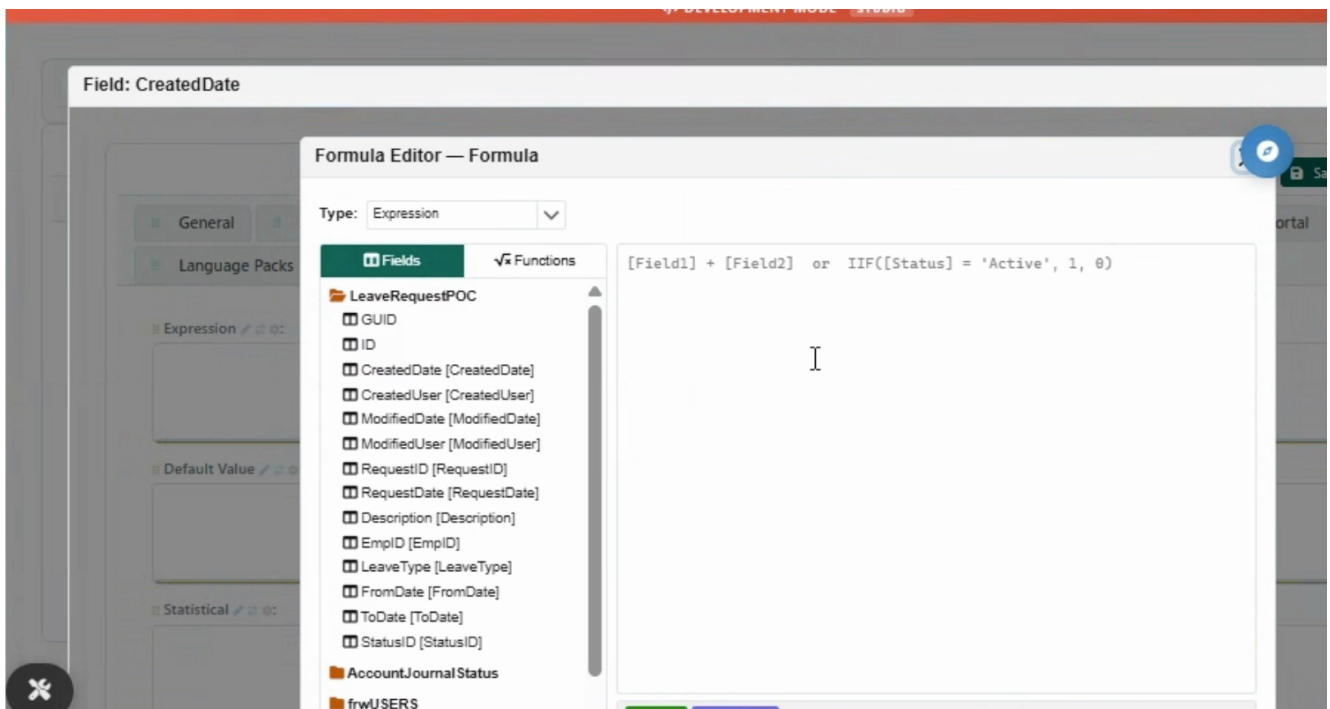


Figure 2. Formula Editor with field and function browsing, about 50 minutes into the recording. Local Test and Validate actions were also observed in the session. [S11]

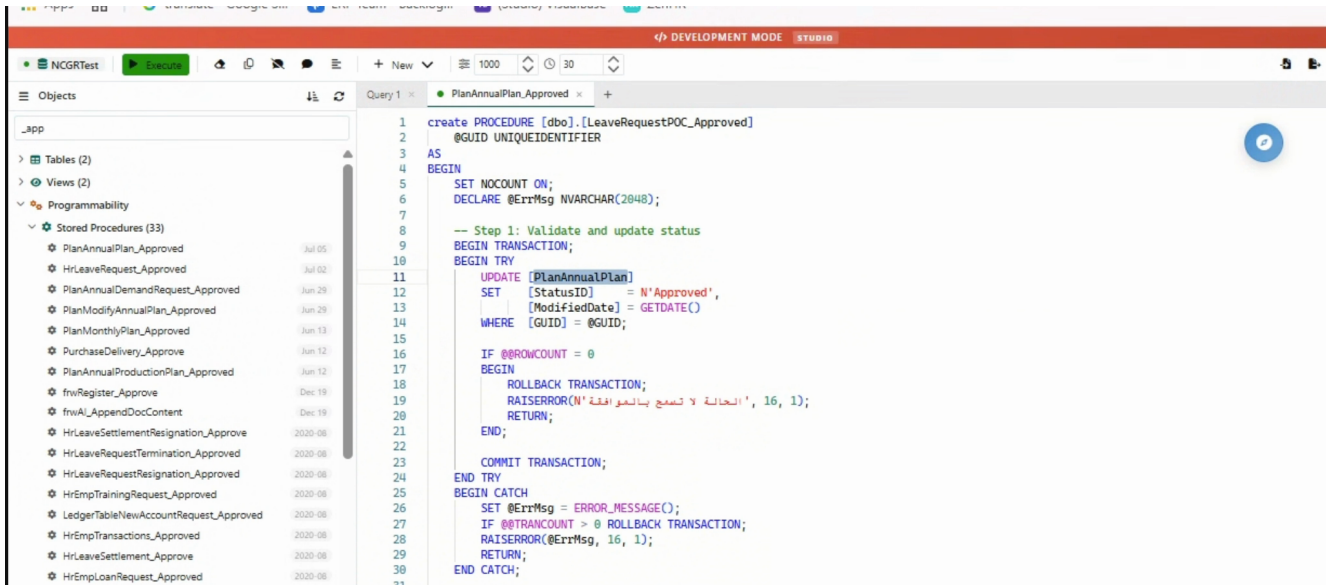


Figure 3. SQL Studio shows an approval procedure being edited, about 72 minutes into the recording. This frame is not evidence of a released defect. [S11]

Interpretation. The recording demonstrates speed and flexibility. It did not show Git review, automated regression tests, a signed release artifact, promotion between environments, or rollback. An unshown control may exist elsewhere; this review cannot credit it without evidence. [S11]

Recording detail. The SQL Studio frame at approximately 72:30 contained an in-progress copied procedure body. It illustrates the manual edit path; it is not reported here as a shipped defect. [S11]

3 The tests the POC did not complete

A screen or configuration option shows an interface or setting, not that its business rule works correctly. For a system of record, NCGR also needs repeatable tests of the rules behind it. The following are acceptance scenarios, not statements that VisualBase cannot support them.

Gate	Plain-language test	Current POC result
Stock correctness	Two users try to issue or reserve the same stock at once. The total must never exceed the available quantity.	Not demonstrated
Retries and reversal	Repeat a receipt or transfer after a timeout. One request must create one business effect; a correction must link to the original.	Not demonstrated
Production history	An order uses an approved bill of materials (BOM) version. Later edits must not silently change what that order used.	Screens visible; rule behavior unverified
Release control	Change a rule, review the full diff, run tests, promote the same version, and show a safe rollback.	Not demonstrated
Security and recovery	Test permissions, restore backups, and recover after a failure against agreed time and data-loss targets.	Not demonstrated

Gate	Plain-language test	Current POC result
Integrations	Show versioned APIs and safe handling of duplicate or retried messages with reconciliation.	Not demonstrated
Units and traceability	Keep unit conversions consistent and trace each serial or lot through receipt, use, inspection, and correction.	Screens visible; rule behavior unverified
Period controls	Reject or govern backdated postings and changes to closed periods; reconcile downstream totals.	Not demonstrated
Audit and correction	Record who changed a rule or posting. Correct a stock movement with a linked reversal, not a silent rewrite.	Options visible; enforcement unverified

Example. If 10 units are available and two users each try to reserve 8 at the same time, the final reserved amount must not become 16. This is a proposed test, not an observed VisualBase failure. For a retried receipt, the same request should create one stock movement even if the sender repeats it after a timeout. These cases should be run against the actual vendor deployment. [S4][S11]

4 Why the architecture matters

A stock rule such as 'do not reserve more than is available' is one business promise. In the assessed model, making or checking that promise may touch more than one place. The diagram is a **conceptual change checklist**, not a measured VisualBase execution trace. [S1][S2][S4][S5]

Places a team may need to inspect for one rule change



Figure 4. Unordered checklist of rule surfaces; their exact call order and dependencies need vendor mapping. [S1][S2][S4][S5]

Question	VisualBase model in this review	Proposed .NET and ABP model
Where do product rules live?	Across a proprietary engine, configuration records, rule tables, SQL views and procedures. [S1][S2][S4][S5]	In NCGR-owned C# modules, with SQL Server retaining persistence, transactions, and constraints. [S6]
How is one change reviewed?	A complete Git-to-release chain was not demonstrated in the POC. [S11]	NCGR would require pull requests, tests, a versioned build, and controlled deployment.
What is already proven?	Fast form and workflow setup was visible. [S4][S11]	No product implementation exists yet. The architecture must still be built and validated.

This is a trade-off, not a claim that stored procedures are always wrong. A tested, versioned SQL procedure may be appropriate for a bounded need. The risk rises when application policy, validation, configuration, and operational state are all changed in the database without a single controlled release path.

Change safety and testability

A developer may be able to test a formula while editing it. That helps with immediate feedback, but it is different from a repeatable test suite that re-runs when another rule, SQL procedure, form, or service changes. NCGR should ask the vendor to show the full change: export every affected artifact, review it with another engineer, run automated tests, promote the same version from test to production, and roll it back. The recording did not demonstrate that chain. [S11]

Database transactions and constraints can protect data; the open question is whether each inventory and production rule has one clear owner and is exercised by concurrency, retry, and failure tests. A good architecture should make the reason for a posting visible and reproduce it during support, including the exact rule and BOM version used. [S4][S11]

Security boundary

The POC showed security settings, but a setting is not proof of enforcement. NCGR needs evidence for who can alter rule tables or SQL procedures, how those changes are approved and logged, whether service identities have least-privilege access, and whether every interface applies the same authorization decision. A stored procedure can be safe when permissions and tests are well designed; a powerful database account or unmanaged runtime edit can bypass intended review. These are tests to run, not discovered vulnerabilities. [S4][S11]

Operations and integrations

A production support team needs to follow one business transaction through the UI, service calls, SQL, and any queue or external system. It needs metrics, structured logs, alerts, backup and restore practice, and a clear owner for each failure. The POC did not provide evidence of those controls across the reported service estate. [S4][S11][S12]

For ERP, warehouse, finance, or other integrations, NCGR should require published API contracts, versioning, authentication, duplicate-message handling, reconciliation, and a way to replay safely after an outage. A successful demo call would not establish these failure behaviors. [S4][S11]

Version and lifecycle

Visualsoft publishes product lifecycle information, but the POC runtime version was not reconciled against it. NCGR should identify the exact deployed platform and service versions, support end dates, upgrade path, and compatibility of custom metadata and SQL. The same discipline applies to the proposed .NET/ABP stack: pin supported versions, plan upgrades, and test them. [S3][S7][S8]

ABP supplies modular .NET application infrastructure. It is not finished inventory software. NCGR must still design the ledger, production rules, authorization, tests, integrations, and support model. The proposed .NET approach would likely require more upfront engineering effort; its ownership benefit depends on NCGR actually building and governing the product code and releases. [S6][S7][S8][S10]

5 Source handover and vendor dependence

The vendor is expected to give NCGR the no-code platform source after project completion. That could improve audit and continuity. The actual agreement, timing, rights, and complete component list remain unverified. [S5]

Source possession is not operational independence. The stakeholder reports more than 20 services and several service-specific frontends. If the agreed handover includes the whole estate and NCGR takes on its support, that would create a sizable integration and maintenance task. A repository is useful only if NCGR or another supplier can build, deploy, monitor, repair, and upgrade the complete estate without vendor-only accounts, undocumented tools, or restricted third-party licenses. Even then, applications still depend on the proprietary runtime and metadata model. The service count, source scope, support ownership, and dependencies must be confirmed. [S1][S2][S5][S12]

A usable handover is a chain of proof



Figure 5. Proposed acceptance sequence. Source delivery alone completes only part of the first step. [S5][S12]

The dependence has five parts: **legal** rights to change and operate; **build** access to every dependency; **architecture** knowledge of connected services; **data and rules** that can be understood and migrated; and **people** who can fix a fault. The vendor can provide source and still leave one of these parts unresolved. NCGR should test all five before treating handover as an exit plan. [S5][S12]

As a **hypothetical test**, a fault spanning a UI, workflow service, and posting service should be traced and repaired end to end; no such incident or call path was observed here. NCGR should request a service-and-frontend inventory with versions, owners, inter-service calls, data stores, secrets, logging, and recovery order. The examples render.visualbase.com and workflow.visualbase.com are stakeholder-provided names, not endpoints this assessment inspected. [S12]

Acceptance test	Evidence NCGR should require
Inventory and rights	Reconcile every service and frontend, its owner, dependencies, APIs, data store, build artifact, license, and source repository. Verify permanent rights to operate, modify, audit, and appoint another maintainer.
Clean build	An NCGR-selected team builds every component from a fresh environment without a vendor workstation, private account, or undisclosed license service.
Deploy and restore	The team deploys the full topology in NCGR infrastructure, restores data and configuration, rotates secrets, and meets agreed recovery targets.
Cross-service defect	The team reproduces an edge case involving multiple services, traces the request, fixes the root cause, adds a regression test, and verifies downstream effects.
Release and rollback	The team produces a controlled multi-service release, observes it in NCGR monitoring, and rolls back safely without vendor intervention.
Upgrade and transfer	The team upgrades a dependency while preserving applications and metadata; an alternative supplier completes a change independently.

Contract recommendation. Put progressive source access, a complete service map, clean-build evidence, runbooks, observability, documentation, training, and independent cross-service repair tests into staged acceptance milestones. Waiting until the project ends would allow dependence to accumulate before NCGR

learns whether the handover is usable. This is a proposed contractual safeguard, not a statement about the current signed terms. [S5][S12]

6 Balanced choice and recommendation

Criterion	VisualBase on current evidence	.NET and ABP as proposed
Early delivery	Strong: forms and workflows were assembled quickly. [S11]	Likely more upfront domain and foundation work; delivery speed was not measured.
Rule visibility	Rules span engine, metadata, and SQL. [S1][S2][S4]	Rules can be written and reviewed in C# modules if NCGR enforces the design. [S6]
Release evidence	End-to-end review, test, promotion, and rollback not shown. [S11]	Must be implemented in NCGR's delivery pipeline; not automatic.
Ownership	Source handover expected after completion; reported 20+ service estate and independent operation unverified. [S5][S12]	Product source can be owned from the first sprint; ABP upgrade and license choices remain.
Long-term skills	Requires platform-specific engine and metadata knowledge.	Uses mainstream .NET and SQL skills, plus ABP knowledge.
Stock and production	Menus and screens visible; concurrency, BOM history, reversal, and ledger behavior untested. [S4][S11]	Domain model can be explicit, but NCGR must design and prove it.
Security	Options visible; effective permissions and cross-service enforcement untested. [S4]	Authorization can be centralized, but requires threat review and negative tests.
Operations	Reported 20+ services; support, recovery, and upgrade path not independently proven. [S12]	A modular monolith reduces initial deployment units; NCGR still owns monitoring and recovery.
Cost and lock-in	Potential early-speed savings; platform/metadata and later handover costs need a priced model.	Likely more upfront engineering effort; ABP edition, licenses, upgrades, and staffing need a priced comparison. [S10]

Proposed product shape

Start with one deployable .NET application divided into clear modules, rather than many separately deployed services from day one. ABP can provide the application foundation. SQL Server remains the transaction database; it is not treated as the only place for business policy. The diagram is a **proposal**, not a built system. [S6]

Proposed inventory-first request path



Figure 6. Proposed request path for the first inventory slice; production and external integration follow later. No .NET product has yet been implemented. [S6]

Module / concern	First explicit responsibility
Catalog and units	Products, sites, units of measure, and controlled conversion rules.
Inventory ledger	Append-only stock movements, balances, reservations, idempotent posting, linked reversals, and reconciliation.
Production	Approved BOM and routing versions, order state, material issue, output, scrap, and traceability.
Access and audit	Role and site permissions, separation of duties, change history, and investigation trail.
Integrations	Versioned contracts, safe retries, an outbox or equivalent reliable delivery, and reconciliation.

A modular monolith is not a promise that every rule sits in one code file. It gives NCGR a smaller initial deployment surface with explicit ownership and release controls. Separate services should be introduced only where scale, team ownership, or operational evidence justifies the cost. [S6]

Recommendation. Seek approval through an architecture decision record for a greenfield .NET 10 LTS and ABP foundation for inventory and production, beginning with a modular monolith and SQL Server. The team should prove one small but real vertical slice: receive stock, post an immutable movement, prevent duplicate effects, reverse a movement, enforce authorization, and reconcile the balance. Do not expand the implementation until correctness, security, performance, recovery, and operational targets pass. [S6][S8]

Reconsideration rule. VisualBase can remain an option if the vendor demonstrates all critical scenarios and NCGR passes the independent source-handover tests. A strong result on speed cannot substitute for a missing correctness, security, recovery, or ownership test.

7 Decision gates and a safe first plan

The choice should be recorded before development starts. The following gates apply to **either** platform. VisualBase must show the controls in its actual deployment; the .NET team must build and prove them. A critical failure should stop production approval rather than be averaged against faster screen delivery.

Critical gate	Evidence to accept
Domain correctness	Pass concurrent stock reservation, retry, reversal, unit conversion, lot/serial trace, BOM-version, and closed-period scenarios with expected ledger totals.

Critical gate	Evidence to accept
Controlled change	Show complete diff, reviewer approval, automated regression results, environment promotion, rollback, and audit of runtime or SQL changes.
Security	Prove role/site separation, least privilege, consistent checks in UI and APIs, secret handling, and negative permission tests.
Operations	Reproduce a cross-component incident; show tracing, alerts, restore drill, recovery targets, and named on-call ownership.
Integration	Publish versioned contracts; prove duplicate and out-of-order handling, outage replay, and balance reconciliation.
Independent ownership	NCGR-selected engineers build, deploy, fix an edge case, test, and upgrade without vendor-only access or know-how.

Proposed sequence. (1) Approve decision and define stock/production rules with business owners. (2) Establish the .NET/ABP foundation, secure delivery pipeline, and observability. (3) Build the inventory vertical slice and run the gate tests. (4) Add production rules and versioned BOM history. (5) Connect external systems and migrate initial data. (6) Run recovery, performance, security, and operational-readiness tests before launch. Each stage should have measured acceptance criteria; this report is not an effort estimate.

For a fair vendor comparison, ask VisualBase to run the same stock and production scenarios on NCGR infrastructure and deliver the corresponding rule diff, test evidence, trace, rollback, and handover result. Record pass, fail, or untested for each gate. Do not score an untested item as zero capability or as a pass.

8 Limits and terms

The review examined the authenticated POC, the development recording, vendor material, stakeholder-provided contract and deployment context, and official .NET/ABP references. It did not inspect the full VisualBase source, a verified inventory and dependency map of the reported services, complete SQL corpus, database permissions, signed agreement, performance at NCGR volumes, penetration-test results, recovery tests, or production support arrangements. The named service UIs are examples from the stakeholder, not independently tested endpoints. [S1]-[S12]

The recommendation is evidence-led and can change with stronger evidence. It should be recorded in an architecture decision record before development starts. The .NET/ABP option also requires its own implementation and acceptance tests; selecting a framework does not make the future product production-ready.

Term	Meaning here
Metadata	Configuration records that tell a platform what screens, fields, or rules to create.
Stored procedure	SQL code kept in the database and executed there.
BOM	Bill of materials: the approved parts and quantities for a product version.

Term	Meaning here
Idempotent	A repeated request has one business effect, even after a timeout and retry.
Modular monolith	One deployed application with clear internal business modules.
Recovery targets	Agreed maximum time to restore service and maximum acceptable data loss.
Architecture decision record	A short signed record of the choice, reasons, evidence, and conditions for revisiting it.

Sources and evidence register

- [S1] Visualsoft, VisualBase product page, vendor description of runtime development and workflow.
<https://visualsoft.com/visualbase/>
- [S2] US8782095B2, Method and system for programming at run-time and dynamic business application, technical model.
<https://patents.google.com/patent/US8782095>
- [S3] Visualsoft, Products Lifecycle Policy, used to identify version-support questions. <https://visualsoft.com/products-lifecycle-policy/>
- [S4] Authenticated NCGR VisualBase POC, directly observed 1 September 2026; access-controlled, not public.
- [S5] Stakeholder-provided contract context: platform source expected after project completion; exact terms not reviewed.
- [S6] ABP official architecture and documentation, modular application framework. <https://abp.io/docs/latest/framework/architecture>
- [S7] ABP official release notes, .NET 10 alignment and current release track. <https://abp.io/docs/latest/release-info/release-notes>
- [S8] Microsoft, .NET and .NET Core Support Policy; .NET 10 LTS active as checked 19 September 2026.
<https://dotnet.microsoft.com/en-us/platform/support/policy/dotnet-core>
- [S9] Saudi National Cybersecurity Authority, Essential Cybersecurity Controls ECC 2-2024; context only, compliance not assessed.
<https://nca.gov.sa/en/regulatory-documents/controls-list/ecc/>
- [S10] ABP official FAQ, free framework and commercial licensing distinction. <https://abp.io/faq>
- [S11] Recorded VisualBase development UX, 80:08, dated 27 August 2026 and reviewed 2 September 2026. Time-coded visual observations above.
- [S12] Stakeholder report from work with the VisualBase team on NCGR-infrastructure deployment, 19 September 2026: more than 20 services and several service-specific frontends, with render.visualbase.com and workflow.visualbase.com given as examples. Topology not independently audited.