Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Test one production operation

Start here · Nikxius design-partner evaluation

Nikxius is execution control and recovery infrastructure for production automation. We test whether **one blocked operation** becomes easier to authorize, execute and recover than with your existing stack.

Start with a **Kubernetes Deployment rollback to one permitted prior image**, tied to an earlier committed image operation. The rollback gets new authority and a separate outcome.

What we install

A customer-controlled Runtime Node, PostgreSQL and a narrowly scoped Kubernetes identity in **one nonproduction environment**. The agent proposes; the Node holds the mutation credential. Your IdP, RBAC/admission, GitOps, monitoring and emergency procedures remain.

A broad agent credential can permit writes outside the task. Nikxius requires that the agent cannot reach an equivalent writer or the Node's credential. We review that boundary together.

What we test

1. Bind exact target, permitted image, authority and current native state.
2. Reject changed actions, stale state and unauthorized retries.
3. Persist dispatch; retain UNKNOWN after a missing response.
4. Restart, reconcile without blind replay, and export evidence.
5. Have your engineer compare the same contract with your native stack.

What we need

A platform owner, security reviewer, disposable target, approved identity/secrets, PostgreSQL and your current execution path. The local fixture needs Node 22+, Docker, kind and kubectl; real customer identity integration is a separate check.

No model key, payment credential, public Runtime endpoint or unrelated production access is required. Support access is agreed explicitly.

The decision

After the proposed four weeks: customer-operated acceptance, measured native comparison and a continuation or exit decision. A committed patch is not application health; UNKNOWN may remain unresolved. This kit grants no production authorization.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Delegate an operation, retain control

Executive overview · Platform, infrastructure and security leaders

A platform team wants automation to change production. The blocked decision is whether the effect can receive exact authority and survive stale state, concurrency, missing responses and recovery.

Identity, RBAC, admission, GitOps and workflow tools may already satisfy the requirement. Nikxius must demonstrate an advantage over that stack to justify another execution component.

The product hypothesis

A customer-controlled Runtime binds authority to an exact operation and independently read native state. It freezes the command, enforces review, records dispatch before I/O and retains outcomes and unresolved obligations. Kubernetes and your IdP remain authoritative.

The first scope is one nonproduction Deployment rollback to a permitted prior image, bound to an earlier committed operation on the same native resource.

Four weeks to a decision

- **Week 1:** agree workflow, owner, native alternative and decision criteria.
- **Week 2:** install identity, credentials and the bounded operator path.
- **Week 3:** compare the same stale-state, lost-response and restart cases.
- **Week 4:** customer engineer runs acceptance; record costs and decision.

Proposed fee: **US$15,000**, split equally at signing and agreed acceptance. This is unvalidated pricing; written scope defines responsibilities, milestones and legal terms.

What establishes value

Value means a meaningful deficiency closed or lower control/recovery burden. Measure installation, custom engineering, operation and support separately. Founder assistance is not customer independence.

Local tests do not establish customer compatibility, production security or willingness to pay. Those are evaluation questions. A committed resource change is not proof of application health.

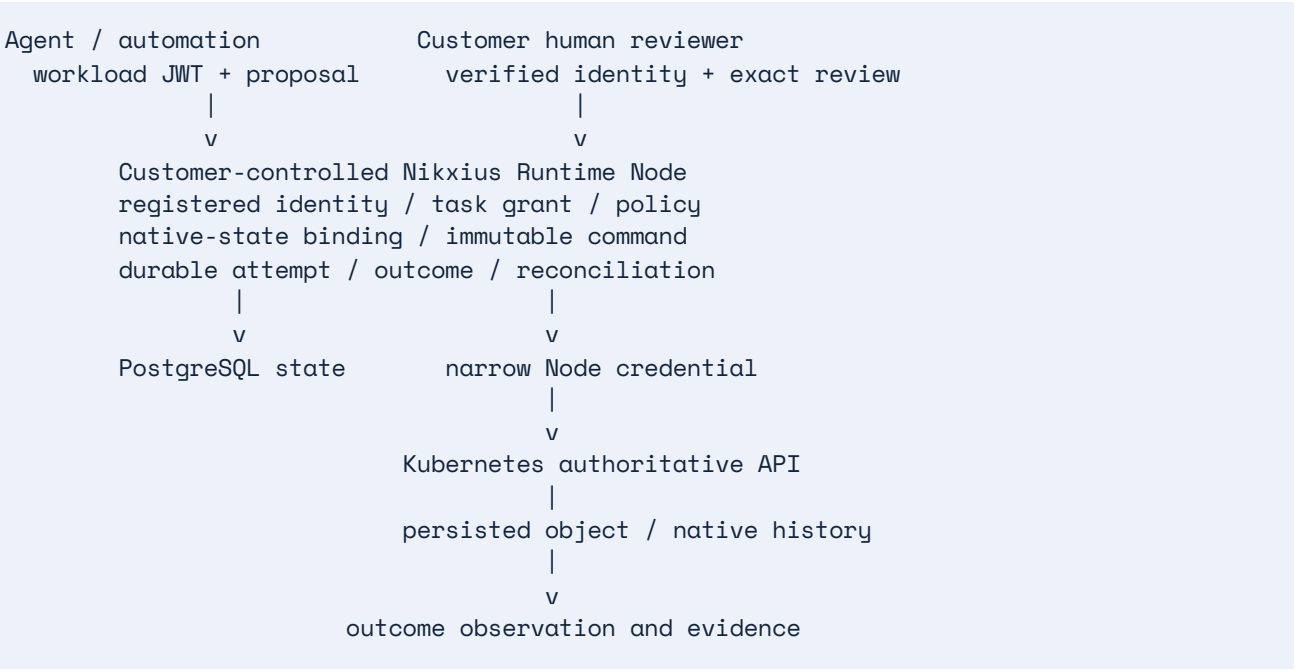
Use the native stack if it meets the contract economically. Continue with Nikxius only for a repeatable advantage. Production authorization and annual terms are separate decisions.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

One operation, one durable record

Technical architecture · Customer-controlled Runtime



Customer IdP supplies verified identities. Customer secret management supplies protected credentials. Native RBAC/admission and existing writer ownership remain authoritative. Monitoring, SIEM and GRC can consume exported records through customer integration; no turnkey SIEM/GRC connector is claimed by this diagram.

Control flow

The workload submits a typed request, grant reference and stable idempotency key. The server derives tenant and principal from registered issuer/subject mappings. It independently reads Kubernetes, checks enrollment and authority, and freezes the target, effect, policy and native UID/resourceVersion. Required human review binds exact command/dispatch hashes and the current operation version.

Before sending a protected mutation, PostgreSQL records admission, reservation and dispatch identity. The Node constructs a conditional native patch. It accepts no arbitrary URL, YAML, shell command or caller-supplied patch. A lost response does not authorize another mutation.

The Node records the persisted native response or retained causal history. Read-only reconciliation can resolve an unknown commitment only with sufficient evidence. A current GET showing the expected image is insufficient on its own.

Read each state dimension

Dimension	Meaning
Lifecycle	`PROPOSED`, `AWAITING_APPROVAL`, `REVALIDATING`, `ADMITTED`, `DISPATCH_RECORDED`, `PENDING`, `UNKNOWN`, `RESOLVED`, `DENIED`.
Commit	`not_dispatched`, `committed`, `rejected`, `unknown`.
Observation	`not_applicable`, `pending`, `converged`, `degraded`, `superseded`, `unobserved`.
Reservation / integrity	Task/resource capacity and separately recorded `unchecked`, `matched` or `mismatch` invariants.

`RESOLVED` does not mean success. `committed` does not mean healthy. A committed invariant mismatch remains an incident with consumed authority.

Integration boundary

The first deployment uses one Node plus PostgreSQL. API and worker may run together. Browser sessions are process-local; the initial evaluation is not a claim of multi-replica availability. PostgreSQL and Kubernetes share no transaction. Worker leases protect local ownership, while the native conditional request protects its frozen baseline.

A GitOps controller that owns the field must be part of the design. Do not create a second uncoordinated writer. If the customer's approved GitOps path already meets the contract, that path is the comparator to beat.

Source: `docs/runtime/API.md`, `OPERATIONS.md`, `DEPLOYMENT.md`, `RECOVERY.md`; `docs/security/TRUST_BOUNDARIES.md` in the evaluation checkout. No hosted Nikxius control plane is required for this Runtime path.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

What the boundary trusts

Security and trust model · Review before granting access

The protected agent is untrusted input to an enrolled operation. The trusted computing base includes the Node and configuration, PostgreSQL and privileged administrators, the configured IdP/JWKS source, Kubernetes API, and customer-controlled admission and credential environment. A compromised Node or cluster administrator is outside the prevention claim.

Identity and credentials

Material	Holder and boundary
Workload JWT	Agent; can propose/inspect its operations. It grants no Kubernetes mutation permission or human role.
Human JWT	Reviewer/sponsor; issuer, audience, registered role, MFA and freshness checked. Synthetic local claims are not real MFA.
Kubernetes token	Node only; mounted read-protected file, reloaded per request, pinned API origin/CA.
Database login	Node uses restricted non-superuser/non-BYPASSRLS role. Separate migration credential is removed after setup.
Evidence key	Optional customer-controlled Ed25519 key file; separate authenticated public-key distribution.

Customer IdP token acquisition is integrated externally. The browser's token handoff is not a full SSO redirect/PKCE client. Identity mappings load at startup; reviewed changes require restart. Task-grant revocation is durable and does not require restart.

Least privilege must include bypass review

Scope the Node to the named resource and required reads/watch/patch. Kubernetes RBAC cannot restrict a patch to one image field; trusted adapter code supplies that restriction. Native admission constraints may add protection and must be tested with the actual controller/admission composition.

The agent must not read Secrets, mint the Node's token, exec into privileged pods, assume another role, invoke an equivalent CI writer or use host access to escape the boundary. The local tests probe named routes. They do not prove that every customer bypass path is absent.

Failure, revocation and break-glass

Missing mandatory authority, policy, approval or durable admission prevents new protected dispatch. That does not recall a request already admitted or stop every delayed sender. Native UID/version/field tests remain necessary. A missing response stays UNKNOWN; no timer converts it to failure.

A Node restart preserves durable operation state in PostgreSQL and ends browser sessions. An old database restore can forget native effects; reconcile the restore gap before reopening writes. Missing native history may leave an outcome unresolved indefinitely.

Customer emergency access remains separately governed. Record the operator, reason, time and native evidence when using it. Never use break-glass to rewrite the original operation as successful, cancelled or nonexistent. The Runtime has no “mark successful” or “reset unknown” endpoint.

Data, evidence and operational responsibility

Operation inputs, identities, decisions, attempts and projected observations stay in the customer’s Node/database. There is no required Nikxius-hosted telemetry or model-provider dependency in this path. Support access and export sharing require explicit scope. Customer infrastructure provides encryption at rest, backups, network controls and retention decisions; their configuration is not established by local tests.

Exports contain supported lifecycle records and stated limits. A signature establishes integrity and attribution under the configured key; it does not establish business correctness, uncompromised execution or independent Kubernetes truth. No automated destructive retention job is supplied; unresolved attempts and consumed identities must not be purged to clear a queue.

No SOC 2, ISO 27001, external penetration test, production SLA, incident-response staffing or customer production acceptance is claimed. Security review is part of the nonproduction evaluation; production needs separate admission.

Source: `docs/security/THREAT_MODEL.md`, `CREDENTIAL_MODEL.md`, `TRUST_BOUNDARIES.md`, `FAILURE_SEMANTICS.md`; `docs/runtime/DEPLOYMENT.md` and `RECOVERY.md`.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Install the evaluation path

Installation guide · Disposable first, customer-controlled next

Use the exact source revision and `docs/runtime/QUICKSTART.md` supplied with your evaluation checkout. That file is the canonical command sequence. The kit does not contain credentials or a separately published Runtime SDK. Runtime code remains private; evaluation access and permitted use are established by the written agreement.

Local prerequisites

Node 22+, npm, running Docker, kind and kubectl; network access to obtain dependencies and pinned sample images. No production cluster, cloud account or LLM key is needed. The helper uses an isolated kubeconfig and named disposable kind cluster, not your current production context.

Run from the repository root. Start with the clean-checkout commands:

```
npm ci
npm run build
```

Continue with the quickstart's **design-partner evaluation section:** isolated Docker database, kind cluster, local migration URL, generated fixture identities and protected environment file. Docker Compose is not required. Supply the migration credential only to setup, then unset it; the Runtime uses its generated restricted login. Setup changes disposable resources; never run it against production.

After setup, run the exact acceptance command:

```
export NIKXIUS_RUNTIME_DEMO_DIR=.tmp/design-partner-eval
node --env-file=.tmp/design-partner-eval/runtime.env \
  scripts/design-partner-acceptance.mjs \
  .tmp/design-partner-eval/runtime-config.json
```

Complete the customer path

Step	Observable result
Configure/start	Strict configuration accepted; `/health` responds and `/ready` reaches the database. Neither checks all external dependencies.
Enroll target	Configuration plus restart, not an enrollment API. The runner generates the exact rollback relationship after the earlier commit.

Step	Observable result
Establish earlier commit	A separately authorized image operation supplies the committed earlier-operation record required for rollback.
Grant/propose/review	Sponsor issues exact rollback grant; workload submits stable key; reviewer inspects current command hashes/version.
Execute/inspect	Operator reads lifecycle, commit and observation separately.
Fault/reconcile	Response loss becomes UNKNOWN; read-only reconciliation uses native evidence or retains uncertainty.
Export	Operation JSON plus optional signature; verify against the separately supplied public key.
Stop/uninstall	Stop the Node and disposable database; retain evidence before explicitly deleting named fixtures.

For operator inspection, serve `runtime-rollback-config.json` as shown in the quickstart and open **http://127.0.0.1:8789/operations.** The fixture's historical `payments` and `production-us-east` labels are synthetic; no production resources are involved. Native conformance suites reset shared fixtures, so run them before final acceptance.

The older public demo uses `set_image`; its fixture reset is not a Runtime rollback. Keep the new run record with its source revision. The sanitized internal verification summary is `artifacts/design-partner-2026-09-15/VALIDATION.json`; generated credentials remain private.

Customer nonproduction installation

Follow `docs/runtime/DEPLOYMENT.md` for image build, migration/init-tenants, registered IdP subjects/audiences, HTTPS origin, mounted rotating credential, reviewed RBAC and Node configuration. The example manifests do not provision IdP, PostgreSQL, ingress, backups or a registry. Customer identity token acquisition and native controller/admission ownership need explicit review.

The **15 September clean-source internal run** reached acceptance in **20 minutes 25 seconds**, including native tests and interleaved author work. Tools were already installed and download caches were warm. The acceptance runner itself took **2 minutes 16 seconds.** These are elapsed internal observations, not hands-on customer installation time. Record prerequisites, downloads and assistance in your run. **Under 60 minutes remains an unvalidated customer target**. See the included evaluation-validation summary for exact timings and limits.

If any step fails, preserve the exact command/version and redacted error. Do not bypass identity, TLS, least privilege or persistence to finish a demonstration. For UNKNOWN, use `docs/runtime/RECOVERY.md`; never mint a new key simply to retry a possibly executed mutation.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Rollback is a new operation

Kubernetes workflow · One Deployment, one permitted prior image

An AI/SRE agent proposes restoring a Deployment container to a prior permitted image. The agent does not receive broad cluster write credentials. Nikxius checks an exact rollback contract; Kubernetes remains authoritative for the conditional mutation and native state.

Bind the relationship

`kubernetes.deployment.rollback_image.v1` requires an explicitly authorized immutable image and `compensatesOperationId`. The earlier operation must be committed in the same tenant, for the same native target/UID, with the appropriate reversed before/after images. An arbitrary old registry digest is not sufficient.

The contract binds the registered cluster, namespace, Deployment, container, native UID/version, image, earlier operation, grant, review policy and stable idempotency key. Customer enrollment must permit the relationship. Use actual observed values; the following is a shape, not an executable grant:

```
{
  "operation": "kubernetes.deployment.rollback_image.v1",
  "target": {
    "cluster": "evaluation",
    "namespace": "evaluation",
    "deployment": "sample-api",
    "container": "api"
  },
  "input": {
    "image": "registry.example.com/sample@sha256:<prior-digest>",
    "compensatesOperationId": "<earlier-committed-operation>"
  },
  "grantRef": "<sponsor-issued-grant>",
  "idempotencyKey": "<stable-task-specific-key>"
}
```

Illustrative tokens above must be replaced from the local runner's actual record. The customer receives a runnable generated example, not these placeholders as an install step.

Review, execute, recover

The Runtime independently reads current state and freezes the exact command. Required approval binds current hashes/version, not "allow a rollback sometime." Before dispatch it revalidates authority and state, persists attempt identity and constructs a conditional patch.

If the native response establishes the intended mutation, record `committed`. Observe rollout separately. If the response is lost, retain UNKNOWN and the reservation; read-only reconciliation must establish the original causal transition. A current matching image or inherited annotation alone cannot establish attribution.

Failure cases to see in the evaluation

Case	Expected boundary
Different target/image/earlier operation	Old grant cannot authorize it.
Resource recreated or concurrently edited	Frozen UID/version cannot silently rebase.
Permission expired/revoked before dispatch claim	New dispatch refused; already accepted writes are not recalled.
Same key, same immutable request	Same operation identity; changed payload conflicts.
Lost response or interrupted worker	Preserve attempt and uncertainty; do not blindly patch again.
Accepted mutation, unhealthy application	Commit remains committed; health and further corrective action require separate assessment.

A rollback may itself fail or remain UNKNOWN. If GitOps owns the image field, use its approved write path or select another valid evaluation target. Source: `docs/runtime/OPERATIONS.md`, `API.md`, `RECOVERY.md` and the acceptance runner supplied with the source checkout.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Acceptance that an operator can repeat

Ten scenarios · Scope, evidence and limitations

Agree the exact nonproduction target, Runtime/source revision, Kubernetes version, identity composition, native comparator, observers and decision owner before testing. Run the same semantic contract on both paths. A skipped or blocked case is not a pass. Preserve redacted operation records, native observations, test output and elapsed operator time.

Required cases

Case	Pass condition	Existing executable evidence
A · Changed action	Old authority/approval cannot approve another target, image or earlier-operation binding.	Runtime A06/A08/A14; rollback adapter/family tests.
B · Expiry/revocation	Expired/revoked authority prevents a later dispatch claim; no false promise of cancelling in-flight native I/O.	A10, including post-claim limits.
C · Stale native state	Changed UID/version/expected field does not receive the old conditional mutation.	A15 native conflict and adapter checks.
D · Duplicate request	Same actor/key/request returns one identity; changed payload or grant conflicts.	A02 actual HTTP/PostgreSQL; response-loss replay.
E · Crash around dispatch	Original attempt/reservation survives; recovery never assumes an unacknowledged transaction rolled back.	A12/A13/A22/A28; actual PostgreSQL restart separately scoped.
F · Lost response	Real possible/committed effect is not blindly repeated.	A16 native response deliberately discarded after I/O.
G · UNKNOWN	Missing sufficient causal evidence remains UNKNOWN, despite a matching current image.	A17/A18 watch-gap and classification checks.
H · Direct bypass	Protected agent cannot perform equivalent protected writes or obtain the Node identity through agreed tested routes.	A24 native permission probes; customer tool/CI/host review still required.

Case	Pass condition	Existing executable evidence
I · Commit vs health	A committed desired-state change remains distinct from rollout observation and application health.	A20; degraded-state injection is labelled.
J · Customer independence	Customer engineer installs, runs, investigates, exports and stops the evaluation using supplied instructions.	**Not established by internal tests. Customer must perform it.**

The **15 September fresh local rollback run passed ten automated assertions across A–I**; J remains **NOT_OBSERVED**. Real Kubernetes 1.35.0 and PostgreSQL were used with synthetic identity/MFA. An actual SIGKILL preserved the original attempt; response-loss recovery retained one adapter PATCH invocation. A separate crash after the durable claim but before I/O preserved UNKNOWN with no replay. These counters are not packet capture or application-effect proof.

The included evaluation-validation summary records **485 distinct repository tests across scoped runs**, without double-counting repeated browser tests. The actual PostgreSQL restart test used a typed native-provider double; the native rollback/SIGKILL exercise is separate. Detailed original A01–A28 mapping remains in `docs/runtime/ACCEPTANCE.md`; these lettered scenarios do not renumber it. Neither run proves customer installation, IdP compatibility or application health.

Run and record

Use the quickstart and customer acceptance runner against disposable resources first. Preserve an earlier committed image operation, then exercise the separately granted rollback. Stop concurrent demo/worker processes when required by the runner; native suites share fixtures and run serially.

For every case record: expected result, actual commit/observation, operation/attempt IDs, grant/command binding, original request key, native observation source, injected fault, repeated-write count, evidence path, operator minutes, assistance and limitations. Fault injection after native I/O is not a claim that every physical network partition was reproduced.

For J, hand a clean environment and the kit to an engineer who did not author the implementation. Count interventions and failed commands. Verify that the engineer can explain why a timeout may remain UNKNOWN and why a new request key is not a safe retry strategy.

Acceptance and purchase are separate

Correct UNKNOWN behavior can pass a test and still impose unacceptable operating cost. Record pass/fail/blocked per case, compare native-stack burden and make a separate continuation decision. Agree the contractual milestone before work; this template creates no automatic acceptance or invoice obligation.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Make the native stack the benchmark

Comparison guide · Same operation, same failure contract

Nikxius earns a place only if it removes a meaningful deficiency or operating cost. Use the customer's strongest credible native path, including the engineering already invested in it. Do not compare a controlled Runtime with a deliberately unrestricted `kubectl` script.

Assemble the real alternative

Existing layer	Include in the comparison
Identity / authorization	Workload identity, token lifecycle, Kubernetes RBAC, named-resource permissions and actual alternate paths.
Native admission	Validating admission policy/webhooks and their exact field/state constraints.
GitOps / CI/CD	Current approved writer, protected review, repository state, controller reconciliation and emergency procedures.
Native API	UID/resourceVersion/field tests, returned persisted object, native events/audit/watch retention.
Workflow / retry	Durable run identity, approvals, dispatch state, retry policy, restart and unresolved outcome ownership.
Evidence / operations	Existing logs, audit exports, dashboards, alerts and operator recovery instructions.

If GitOps owns the field, assess the source-of-truth change and resulting controller effect. A direct patch demonstration cannot claim equivalence merely because both eventually show the same image. Record differences in the declared operation and attribution boundary.

Test identical questions

Can old approval authorize changed bytes? What prevents stale-state mutation? What survives a crash? Who determines whether retry is safe after a timeout? What evidence establishes the original effect? Can an agent bypass the path? What remains unresolved after an unhealthy rollout?

Apply acceptance cases A–J to both approaches, documenting cases that are not semantically comparable. Preserve errors and unresolved outcomes. Native capability that meets a case is a pass, not something to discount because it lives across several tools.

Measure effort honestly

Measure	Capture for each path
Setup	Prerequisites, configuration minutes, custom integration time and people involved.
Code/config	Changed non-generated lines and files, plus existing systems reused. Lines alone are not quality.
Operation	Approvals, manual interventions, normal-run time and incident/recovery minutes.
Failure	Duplicate native writes, stale-action acceptance, missing evidence, duration and ownership of UNKNOWN.
Ongoing burden	Credential rotation, upgrades, retention, backups, monitoring and support dependency.

Time both from the same start condition. Separate founder implementation, customer configuration, custom engineering and recurring operation. Publish neither invented savings nor a composite score that hides an unacceptable failure. The checkout's `docs/design-partner/NATIVE_STACK_COMPARISON.md` contains the reproducible comparator and run-evidence references; **customer-specific savings and superiority remain unmeasured** until the customer baseline is run.

Decide

The 15 September local native reference passed **six component checks**, including native authorization/admission, conditional writes, revocation and response-loss recovery from retained history. Its single-process file journal is not a complete distributed workflow, GitOps or customer-identity composition. The result demonstrates useful native capability; it does not establish the cheaper complete customer path. See the included native comparison reference and its `result.json` for the exact scope.

Choose the native stack if it meets the contract economically. Continue with Nikxius only when the customer records a concrete advantage, accepts its additional operational component and identifies a production decision and budget owner. If the path cannot be made mandatory, the initial boundary is not established.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

One workflow. One environment. One decision.

Proposed four-week design-partner evaluation

Proposal for discussion; not an executed agreement, invoice or customer-system authorization. The price and buying motion remain commercial hypotheses. Customer-specific scope, parties and legal terms require written agreement before work begins.

Scope and commercial structure

One `kubernetes.deployment.rollback_image.v1` operation family: one enrolled Deployment container in one named nonproduction environment, restoring an explicitly permitted immutable prior image tied to an earlier committed operation. Establishing the controlled prior image operation is part of the test fixture, not a license for general cluster mutation.

Item	Proposed term
Duration	Four weeks after written scope, prerequisites and owners are confirmed.
Fee	US$15,000; tax treatment and invoice deadlines agreed in the signed order.
Signing installment	US$7,500.
Acceptance installment	US$7,500 at the milestone agreed before work starts.
Annual credit	Fees actually paid, up to US$15,000, credited to the first annual subscription signed within 90 calendar days of recorded evaluation completion.

Annual scope and price are separate. The proposed credit is not a cash refund, automatic renewal or obligation to purchase. Access delays and scope changes require a recorded schedule decision; no unlimited extension or engineering is implied.

Delivery and responsibilities

Week	Joint outcome
1	Confirm blocked workflow, current controls, native baseline, access, security reviewer and decision criteria.

Week	Joint outcome
2	Install the bounded operation, identity/grant/review integration and operator path.
3	Run stale-state, duplicate, crash and response-loss cases on both approaches; record recovery burden.
4	Customer engineer runs acceptance; deliver findings, evidence and a dated continuation/native-stack decision.

Nikxius, led by Nikhil Sood, supplies installation/integration assistance, the agreed test procedure, documented findings and operator handover. Support channel, working hours, escalation owner and response expectations are recorded in the order; 24/7 coverage and an SLA are not included by implication.

The customer supplies a platform owner, technical operator, security reviewer, approved nonproduction environment, identity/secret/database setup, current-stack configuration and operational evidence. Secrets remain in approved channels. The customer retains change authority, backups and emergency responsibility.

Acceptance and end state

Before signing, name the milestone owner, ten-case acceptance contract, evidence and any measured improvement threshold. An honest unresolved outcome can be technically correct; it must still be operationally acceptable. Disputed/unmet criteria, remedies and termination follow the signed agreement, not assumptions in this offer.

Close with one dated choice: annual/production scoping, additional bounded evaluation, native stack sufficient, or no continuation. Production access, new operations, bespoke integrations, compliance certification and public customer references require separate approval. No endorsement or right to publish customer data is implied.

Contracting identity, authorized signatories, license, confidentiality, data handling, IP, liability, tax, termination and governing law remain to be confirmed through appropriate legal review. This scope supersedes the older image-change starting offer for new evaluations while retaining its proposed price and credit structure.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Questions before an evaluation

Technical FAQ · Scope and honest limits

Why not use RBAC and admission?

Use them. They are part of the baseline and the Nikxius deployment. RBAC constrains the credential's resource/verbs; native admission can constrain writes. Evaluate whether the composed stack also binds the exact task, native baseline, durable attempt, unresolved outcome and operator recovery economically. Nikxius has no automatic claim to win.

Does this replace GitOps, Temporal or the IdP?

No. Existing identity, workflow and authoritative writer ownership remain. The current typed adapter calls the registered Kubernetes API; it is not a generic GitOps or Temporal integration. A competing controller must be reconciled in the design, not ignored during the demo.

Is this a published SDK or hosted service?

The Node client is exported by the private control-plane workspace. Runtime source is private and UNLICENSED without agreed evaluation terms. Execution requires no Nikxius SaaS control plane. The public website is not a customer execution endpoint.

Can the agent still bypass it?

Yes, if it retains another equivalent writer or access to the Node's credential. Review direct permissions, secrets, token creation, exec, CI/tools and host paths. Local denial tests establish the tested paths, not every possible customer route.

What if the response is lost?

The attempt may have executed. Preserve UNKNOWN and the original identity; use read-only reconciliation. Do not issue another mutation simply because a timeout occurred. Sufficient retained native history can establish the original transition; current matching state alone cannot.

Can UNKNOWN stay forever?

Yes. Missing or compacted history can prevent resolution. The task/resource reservation remains. Customer operational ownership and separately governed emergency access must handle that obligation. A UI acknowledgement does not manufacture evidence.

Is execution exactly once?

There is no universal exactly-once claim. Nikxius preserves request/attempt identity, avoids blind replay and uses native preconditions. PostgreSQL and Kubernetes are separate systems; delayed requests and missing history remain explicit limits.

Does revocation cancel an action?

It blocks later checked admission/dispatch claims. It does not retract a native request already accepted or guarantee every delayed sender has stopped. The exact native conditional request is a separate guard.

Does a signature prove a correct outcome?

It can establish integrity and Node-key attribution of the retained record. It does not establish business correctness, independent native truth or a complete record where history was lost. Unsigned source-qualified exports remain possible.

What proves that the application recovered?

Application-specific health checks owned by the customer. A committed Deployment spec change and controller rollout observation are distinct from application health. The rollback is itself a new consequential operation and can fail or remain unknown.

Is it production-ready or independently certified?

Local conformance and this proposed nonproduction evaluation do not establish production acceptance, external penetration testing or certifications. Customer IdP, native policy, backups, retention, support and security review must be validated for the intended environment.

Nikxius

DESIGN-PARTNER EVALUATION · PROPOSED

Agree the decision before the test

Success and exit criteria · Technical acceptance is one gate

The evaluation is useful only if it changes a real decision. Record the blocked operation, platform owner, technical owner, budget owner, current alternative, production consequence and decision date before starting. Interest in AI alone is insufficient.

Technical gate

The customer engineer can install the agreed composition, establish exact authority, run the supported rollback, observe failure, preserve UNKNOWN, reconcile using native evidence and export the lifecycle. Cases A–J have explicit pass/fail/blocked evidence and limitations. No bypass is found within the agreed review scope. Founder interventions and unresolved security issues are counted.

A correctly retained UNKNOWN may satisfy a failure-semantics test. It does not by itself establish acceptable day-to-day operations. Define who investigates unresolved effects, available history, acceptable investigation time and escalation before acceptance.

Economic and operating gate

Compare the same contract with the actual native alternative. Measure customer installation/configuration, custom engineering, support and incident/recovery effort separately. Agree a useful improvement threshold with the owner instead of imposing an invented percentage. Include ongoing Node/database upgrades, credential rotation, backups, monitoring and retention.

Recurring software value requires substantially reusable contracts. If each customer needs different engineering and continual founder operation, record the result as services-heavy rather than calling it self-service software.

Four closeout outcomes

Decision	Evidence required
Continue to annual/production scoping	Technical acceptance, useful native-stack advantage, named budget owner and separately planned production admission.
Extend a bounded evaluation	Specific unresolved question, owner, time/cost cap and written scope decision.
Native stack sufficient	It meets the agreed contract at acceptable operating cost. Preserve that finding.

Decision	Evidence required
Stop	No urgent workflow, no repeatable advantage, unacceptable trust/availability burden or no willingness to pay.

The acceptance installment uses the previously agreed contract milestone. Purchase, payment, technical acceptance and permission to change production are separate records. A technical pass with no willingness to pay remains an unproven commercial thesis.

Close access and retain the truth

Export the agreed redacted result and lifecycle evidence. Revoke evaluation grants and remove temporary identities/credentials through customer procedures. Stop/delete only explicitly agreed evaluation resources after retaining required evidence; do not silently destroy unresolved obligations. Record remaining UNKNOWN operations and transfer ownership before shutdown.

Document next operation requests as evidence, not automatic roadmap promises. Broader implementation should normally follow repeated demand from independent companies with the same missing requirement and a budget owner, unless a paying customer explicitly funds a bounded bespoke scope.

The closeout records the completion date, customer decision, reasons, assistance hours and data/access disposition. That date anchors any agreed 90-day annual-credit period. Neither public reference rights nor customer endorsement is included by default.