SOLA Generated Component Contracts

Java source and requirement annotation traces

Contents

Cell **2**
- Boundary . . . 2
- Requirements . . . 2
  - R1: Health . . . 2
  - R2: Manifest . . . 2
- Out of scope . . . 3

Public documentation **3**
- Boundary . . . 3
- Requirements . . . 3
  - R1: Public access . . . 3
  - R2: Confinement . . . 3
  - R3: Evidence . . . 3
- Decisions . . . 3

History **3**
- Boundary . . . 3
- Requirements . . . 3
  - R1: History . . . 3
- Out of scope . . . 3

Identity **4**
- Boundary . . . 4
- Requirements . . . 4
  - R5: First factor . . . 4
  - R6: External proof . . . 4
  - R7: Grants . . . 4
- Decisions . . . 4

Iteration **4**
- Boundary . . . 4
- Requirements . . . 5
  - R1: Submit iteration . . . 5
  - R2: List iterations . . . 5
  - R3: Run iteration . . . 5
- Entities . . . 5
- Decisions . . . 5
- Out of scope . . . 5

Sola zero **6**
- Components and permitted collaboration . . . 6
- Stack . . . 6

System invariants . . . 6
References . . . 6

Requirement annotation traces **6**
sola.cell:R1.1 . . . 6
sola.cell:R2.1 . . . 6
sola.documentation:R1.1 . . . 7
sola.documentation:R1.2 . . . 7
sola.documentation:R2.1 . . . 7
sola.documentation:R3.1 . . . 7
sola.history:R1.1 . . . 7
sola.history:R1.2 . . . 8
sola.identity:R5.1 . . . 8
sola.identity:R5.2 . . . 8
sola.identity:R5.3 . . . 8
sola.identity:R6.1 . . . 8
sola.identity:R6.2 . . . 9
sola.identity:R6.3 . . . 9
sola.identity:R6.4 . . . 9
sola.identity:R6.5 . . . 9
sola.identity:R7.1 . . . 9
sola.iteration:R1.1 . . . 10
sola.iteration:R1.2 . . . 10
sola.iteration:R1.3 . . . 10
sola.iteration:R1.4 . . . 10
sola.iteration:R1.5 . . . 10
sola.iteration:R2.1 . . . 11
sola.iteration:R3.1 . . . 11
sola.iteration:R3.2 . . . 11
sola.iteration:R3.3 . . . 11
sola.iteration:R3.4 . . . 11
sola.iteration:R3.5 . . . 11
sola.iteration:R3.6 . . . 12
sola.iteration:R3.7 . . . 12
sola.iteration:R3.8 . . . 12

Generated documentation is a projection of source declarations, not a conformance or proof certificate.

Cell

Identify this cell and its installed capabilities.

Boundary

- `read-health` — report process availability.
- `read-manifest` — report this cell's identity and capabilities to its owner.

Requirements

R1: Health

- R1.1 — When asked for health, cell shall report availability without owner data.

R2: Manifest

- R2.1 — When an owner requests the manifest, cell shall identify itself as a zero cell with no business modules.

Out of scope

Fleet administration and deployment orchestration.

Public documentation

Serve the generated contracts of this cell composition without a login.

Boundary

- `home` — present this named cell and its generated documentation.
- `artifact` — serve generated HTML, PDF, JSON and published source files.
- `manifest` — identify the documentation build and its evidence.

Requirements

R1: Public access

- R1.1 — When a visitor requests the homepage without credentials, documentation shall serve the generated homepage.
- R1.2 — When a visitor requests a published document without credentials, documentation shall serve its content and declared media type.

R2: Confinement

- R2.1 — If a requested path escapes the publication root or names an unpublished file, documentation shall return not found.

R3: Evidence

- R3.1 — When a visitor requests the manifest, documentation shall identify its build and distinguish annotation links from test and proof evidence.

Decisions

D1 — Generation happens in the reviewed build pipeline. HTTP requests only read its artifacts. D2 — This brick is public; the zero cell's owner and authentication contracts remain unchanged. D3 — Only reviewed source and synthetic verification summaries are published. Runtime state and credentials are outside the publication root.

History

Let an owner inspect the durable history of their requests.

Boundary

- `list-history` — read the owner's operational events.

Requirements

R1: History

- R1.1 — When an iteration is accepted, history shall expose an event identifying its actor and request.
- R1.2 — When an owner requests history, history shall exclude other owners' events.

Out of scope

Prompt content, credentials, process output and telemetry recordings.

Identity

Exchange the owner's key and an external approval for a cell-local bearer grant. Implements key-and-factor-v1. Matrix transport lives in the optional matrix-login-factor brick.

Boundary

- `challenge` — prove the owner key and obtain a challenge plus a private redemption secret.
- `pending` — let the configured factor adapter discover pending challenges.
- `prove` — let that adapter attest the owner's approval or denial.
- `redeem` — return a grant only to the initiating client after approval.
- `owner`, `logout` — identify and revoke the presented grant.

Requirements

R1–R4 belonged to the retired magic-link prototype. They are not reassigned.

R5: First factor

- R5.1 — Unless an unexpired grant is supplied, identity shall refuse owner data access; neither factor credential shall act as a grant.
- R5.2 — When the configured owner key is presented, identity shall issue a short-lived challenge and a private redemption secret.
- R5.3 — While an unexpired challenge is pending, identity shall refuse another pending challenge for that owner.

R6: External proof

- R6.1 — Only after the configured factor approves a challenge shall identity exchange its private redemption secret for one grant.
- R6.2 — If approval is denied, identity shall refuse redemption.
- R6.3 — If the challenge expires, identity shall refuse proof and redemption.
- R6.4 — If the proof names another owner, was already used, or targets a decided challenge, identity shall reject it.
- R6.5 — When redemptions race, identity shall issue at most one grant.

R7: Grants

- R7.1 — Identity shall persist only grant/redemption hashes, expire grants, and revoke a grant on logout.

Decisions

D1 — The factor adapter is trusted to attest Matrix identity. Room membership is not authorization. D2 — A grant grants the configured owner access to this cell only. It has no power on another cell. D3 — A key and Matrix account are two factors; a reaction alone is not device verification or phishing resistance. D4 — The first prototype's magic-link routes and cookies are removed. A UI can implement the same challenge exchange.

Iteration

Accept an owner's change request and record its execution outcome.

Boundary

- `submit-iteration` — accept a prompt, optionally requesting execution.
- `list-iterations` — read the owner's change requests.
- `run-iteration` — request execution of a queued change through the configured runner.
- `read-iteration` — inspect one request, its outcome and verified receipt.

- `cancel-iteration` — cancel queued or running work.
- `read-patch` — retrieve the verified candidate belonging to this owner.

Requirements

R1: Submit iteration

- R1.1 — When a nonblank prompt is submitted, iteration shall durably queue it with an identifier and creation time.
- R1.2 — If the prompt or execution options are invalid, then iteration shall reject the request before creating a record.
- R1.3 — Unless execution is requested, iteration shall leave the request queued without invoking a runner.
- R1.4 — If source changes or archives are supplied, then iteration shall refuse them before creating a record.
- R1.5 — When a submission repeats an idempotency key and payload, iteration shall return the original request; a changed payload shall conflict.

R2: List iterations

- R2.1 — When an owner lists iterations, iteration shall return only that owner's requests.

R3: Run iteration

- R3.1 — If no runner is configured, then iteration shall refuse execution without consuming the queued request.
- R3.2 — When execution succeeds, iteration shall record completion without claiming deployment.
- R3.3 — If execution fails or exceeds its time limit, then iteration shall record `failed` or `timed-out` respectively.
- R3.4 — If the request is unknown, belongs to another owner or has already started, then iteration shall refuse execution.
- R3.5 — Where flight recording is enabled, iteration shall record the run's identifier, duration and outcome without the prompt or credentials.
- R3.6 — When cancellation is accepted, iteration shall stop its process group and retain cancellation despite a racing completion.
- R3.7 — After a process restart, iteration shall stop identified orphan runners, mark running requests interrupted, and preserve queued execution intent without replaying interrupted work.
- R3.8 — Only after the configured verifier returns a matching receipt and patch hash shall iteration report completion and expose that patch to its owner.

Entities

- CandidateReceipt — request, source base commit, patch hash, verification and deployment flags.
- RunEvent — opt-in JFR timing without private payload.

Decisions

D1 — One metal worker consumes requests serially. SQLite ownership is exclusive per process. D2 — A runner process receives JSON, never a shell command derived from a prompt. D3 — Successful execution produces a reviewable candidate. Applying and deploying it is a separate operation. D4 — The configured verifier is trusted. Workspace separation does not provide OS isolation.

Out of scope

Agent selection, tool-call/token accounting, patch upload and automatic deployment.

Sola zero

An authenticated owner submits a change and inspects a verified candidate.

Components and permitted collaboration

- HTTP boundaries parse credentials and invoke identity control before owner operations.
- Identity and iteration record events through history control in their own transactions.
- Identity, iteration and history own their schemas and use store for connection mechanics.
- Runner invokes iteration control; it does not call an HTTP boundary.
- Cell publishes metadata. Matrix, Codex, tap leasing and repository verification are external bricks.

Stack

Java 25, Quarkus REST, SQLite on metal or Aurora DSQL on AWS. The single process runner is supported on Linux metal only. Lambda queues work without running it.

System invariants

- S1 — The cell shall reject unauthenticated reads of owner data.
- S2 — The cell shall keep one owner's iterations and history invisible to other owners.
- S3 — When the process restarts, the cell shall retain accepted iterations, grants and history.

References

- SBCE.
- Adam Bien's ebank component example.

Requirement annotation traces

Generated from canonical package contracts and resolved Java annotations.

A link declares intended coverage. This report neither runs tests nor proves correctness.

sola.cell:R1.1

When asked for health, cell shall report availability without owner data.

Implementation links:

Missing

Test links:

Missing

sola.cell:R2.1

When an owner requests the manifest, cell shall identify itself as a zero cell with no business modules.

Implementation links:

Missing

Test links:

Missing

sola.documentation:R1.1

When a visitor requests the homepage without credentials, documentation shall serve the generated homepage.

Implementation links:

- DocumentationResource.home — source line 21

Test links:

- DocumentationTest.publicHomepage — source line 27

sola.documentation:R1.2

When a visitor requests a published document without credentials, documentation shall serve its content and declared media type.

Implementation links:

- DocumentationResource.artifact — source line 26

Test links:

- DocumentationTest.publicArtifact — source line 29

sola.documentation:R2.1

If a requested path escapes the publication root or names an unpublished file, documentation shall return not found.

Implementation links:

- DocumentationResource.artifact — source line 26

Test links:

- DocumentationTest.refusesUnpublishedAndEscapes — source line 31

sola.documentation:R3.1

When a visitor requests the manifest, documentation shall identify its build and distinguish annotation links from test and proof evidence.

Implementation links:

- DocumentationResource.manifest — source line 30

Test links:

- DocumentationTest.publicEvidence — source line 36

sola.history:R1.1

When an iteration is accepted, history shall expose an event identifying its actor and request.

Implementation links:

Missing

Test links:

Missing

sola.history:R1.2

When an owner requests history, history shall exclude other owners' events.

Implementation links:

Missing

Test links:

Missing

sola.identity:R5.1

Unless an unexpired grant is supplied, identity shall refuse owner data access; neither factor credential shall act as a grant.

Implementation links:

- IdentityResource.owner — source line 41

Test links:

- ZeroTest.factorsAreNotAccessTokens — source line 49

sola.identity:R5.2

When the configured owner key is presented, identity shall issue a short-lived challenge and a private redemption secret.

Implementation links:

- IdentityResource.challenge — source line 20

Test links:

- ZeroTest.challengeRequiresFirstFactor — source line 56

sola.identity:R5.3

While an unexpired challenge is pending, identity shall refuse another pending challenge for that owner.

Implementation links:

- IdentityResource.challenge — source line 20

Test links:

- ZeroTest.onlyOnePending — source line 62

sola.identity:R6.1

Only after the configured factor approves a challenge shall identity exchange its private redemption secret for one grant.

Implementation links:

- IdentityResource.redeem — source line 35

Test links:

- ZeroTest.requiresBothProofs — source line 68

sola.identity:R6.2

If approval is denied, identity shall refuse redemption.

Implementation links:

- IdentityResource.redeem — source line 35

Test links:

- ZeroTest.denial — source line 81

sola.identity:R6.3

If the challenge expires, identity shall refuse proof and redemption.

Implementation links:

- IdentityResource.prove — source line 29
- IdentityResource.redeem — source line 35

Test links:

- ZeroTest.expiry — source line 87

sola.identity:R6.4

If the proof names another owner, was already used, or targets a decided challenge, identity shall reject it.

Implementation links:

- IdentityResource.prove — source line 29

Test links:

- ZeroTest.replayAndSubject — source line 95

sola.identity:R6.5

When redemptions race, identity shall issue at most one grant.

Implementation links:

- IdentityResource.redeem — source line 35

Test links:

- ZeroTest.concurrentRedemption — source line 105

sola.identity:R7.1

Identity shall persist only grant/redemption hashes, expire grants, and revoke a grant on logout.

Implementation links:

- IdentityResource.logout — source line 46

Test links:

- ZeroTest.grantLifetime — source line 116

sola.iteration:R1.1

When a nonblank prompt is submitted, iteration shall durably queue it with an identifier and creation time.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R1.2

If the prompt or execution options are invalid, then iteration shall reject the request before creating a record.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R1.3

Unless execution is requested, iteration shall leave the request queued without invoking a runner.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R1.4

If source changes or archives are supplied, then iteration shall refuse them before creating a record.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R1.5

When a submission repeats an idempotency key and payload, iteration shall return the original request; a changed payload shall conflict.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R2.1

When an owner lists iterations, iteration shall return only that owner's requests.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.1

If no runner is configured, then iteration shall refuse execution without consuming the queued request.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.2

When execution succeeds, iteration shall record completion without claiming deployment.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.3

If execution fails or exceeds its time limit, then iteration shall record `failed` or `timed-out` respectively.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.4

If the request is unknown, belongs to another owner or has already started, then iteration shall refuse execution.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.5

Where flight recording is enabled, iteration shall record the run's identifier, duration and outcome without the prompt or credentials.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.6

When cancellation is accepted, iteration shall stop its process group and retain cancellation despite a racing completion.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.7

After a process restart, iteration shall stop identified orphan runners, mark running requests interrupted, and preserve queued execution intent without replaying interrupted work.

Implementation links:

Missing

Test links:

Missing

sola.iteration:R3.8

Only after the configured verifier returns a matching receipt and patch hash shall iteration report completion and expose that patch to its owner.

Implementation links:

Missing

Test links:

Missing