**1. Overview**

* **Title:** HIPPO Design
* **Created by:** Shelly (TECH)
* **Last modified:** Dec 05, 2024 (as per document, likely a placeholder)
* **Design Status:** APPROVED
* **Development Status:** IN PROGRESS
* **Owner:** @Arroyo, John (TECH)
* **Architect:** @Guledi, Shelly (TECH)

**2. What is GATR?**  
GATR (Quality Gates) is a pipeline-agnostic platform designed to implement quality gates at various phases of the software delivery lifecycle. It functions as an ecosystem of APIs and a UI, powered by a rule engine (CPA and Drools) in the backend.

* **Key Features:**
  + Allows users with appropriate permissions to configure rules, conditions, and actions within the rule engine based on scope.
  + Enables implementation of automated pipeline controls without requiring changes to existing code or platform scripts.
  + Features an appropriate break-glass opt-in/opt-out mechanism.
  + Integrates with CITI policies, CSRA (Cyber Security Risk Appetite), architectural governance, and other external data sources to detect deviations.
  + Ensures CITI compliant and high-quality code progresses into production.

**3. GATR Principles**

1. **Pipeline Agnostic:** GATR is compatible with all pipelines.
2. **Quality by Design:** Automatically measures metrics and safeguards against poor quality.
3. **Actionable Feedback:** Increases developer productivity by providing human-readable failure reasons and remediation tips where applicable.
4. **Gating Inspection:** Provides Realtime APIs to generate documentation and perform inspections at any given instant.
5. **Consistency:** Standardized development and implementation for a streamlined user experience.
6. **Observability and Stability:** A performant and stable ecosystem.

**4. LSE CI (Lightspeed Enterprise Continuous Integration)**

* **Architecture:**
  + A developer pushes code to Bitbucket, triggering the LSE CI workflow.
  + The workflow includes: CI Clone, Application Build, Trigger Scan (e.g., Snyk).
  + Scan results are pushed to an S3 bucket and a Kafka Topic.
  + A "Snyk Scan Consumer" (part of Gatr Components) consumes messages from Kafka, downloads full scan results from S3, and stores them in the "Gatr DB" (MongoDB).
  + The LSE CI Workflow's "Process Gates" step makes a POST /ci/gates call to the "Gatr REST API".
  + The Gatr REST API processes the gates against rules in Gatr DB and returns a summary.
  + The LSE API stores this summary in the "LSE DB".
  + The LSE UI allows users to view build summary and detail screens by fetching data from LSE DB and, for detailed gate results, by making a GET /ci/results/{gatrId} call to the Gatr REST API.
* **Notable Points (LSE CI):**
  + **Authentication (Phase 2):** Basic authentication with fine-grained passwords. COIN will be integrated later.
  + **Retention Period:** 8 days for S3 storage and GATR DB.
  + **Pre-signed URL:** Accessible for up to 7 days.
  + **Downstream API Failure:** If any downstream API call from Gatr fails, the gate result will be PASS.
  + **Snyk.com Results (Phase 2):** Only uploaded to S3. With Snyk Annotations API integration, a compressed format will be uploaded.
  + **API SLA:** 20 seconds, with a timeout after 20 seconds.
  + **S3 URL:** Subject to change; GATR side will be informed if changes are needed.
  + **Race Condition (Snyk Scan):** Sufficient time must be ensured between Snyk scan result availability and the CI gates call. If results are missing, the gate will PASS.
  + **Policykit:** Out of scope. If multiple "slooms" (Security Level Objective and Outcome Metrics) exist, GATR will consider the first "sloom" for decision-making.
* **Workflow (LSE CI):**
  + Developer checks in code, triggering the build pipeline (clone, compile, build, trigger scan).
  + Scan results (metadata to Kafka, full results to S3) are pushed.
  + Snyk Scan Consumer monitors Kafka, downloads from S3, and stores in gatrscanresults Mongo collection.
  + Pipeline calls /ci/gates endpoint to process gates and get a summary.
  + Gate execution summary is stored in LSE database.
  + Build status (failed/successful) is determined based on Gatr's overall result.
  + Developers view gate summary in LSE UI (sourced from LSE DB).
  + Detailed gate results are fetched via the /ci/results endpoint.
* **Gate URLs (LSE CI):**
  + POST /ci/gates: Process ci gates for LSE. (Auth: AD)
  + GET /ci/results/{gatrId}: Fetch results for LSE ci gates. (Auth: Open)
  + POST /ci/data: Upload scan results (UNIFY / LSC / ICGBUILD). (Auth: AD)
  + URLs provided for DEV, UAT, PROD environments (e.g., https://<env>-cicd-gatr.nam.nsroot.net/ci/gates).
* **API Specifications (LSE CI):**
  + **1. Process CI Gates (POST /ci/gates)**
    - **Request Body:**
    - {
    - "csild": 164603,
    - "project": "fin-unify",
    - "pipelineName": "fin-d-cicd-unify-archctr",
    - "buildVersion": "57b-45f4a60-feature-sdlc",
    - "pipelineType": "lse",
    - "branch": "master",
    - "branchType": "release", // or "nonRelease", mandatory in future
    - "buildId": "", // optional, Loki buildId
    - "buildType": "loki", // mandatory if buildId present
    - "buildInstance": "" // optional, Loki instance

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* + - **Field Documentation:** csild (mandatory, int), project (mandatory, string), pipelineName (mandatory, string), buildVersion (mandatory, string, LSE value), pipelineType (mandatory, string, "lse"), branch (mandatory, string), branchType (optional, "release"/"nonRelease", will be mandatory), buildId (optional, string, Loki build ID), buildType (optional, string, "loki" if buildId present), buildInstance (optional, string, Loki instance if buildId present).
    - **Successful Response (200 OK):**
    - {
    - "status": "SUCCESS",
    - "gatrId": "GATR-6dd1290c-2552-4e51-9174-e652666d3362",
    - "csild": 164603,
    - "project": "fin-unify",
    - "pipelineName": "fin-d-cicd-unify-archctr",
    - "buildVersion": "710-011e3d-master",
    - "stage": "CI",
    - "pipelineType": "LSE",
    - "overallResult": "PASS", // PASS, FAIL, WARN, PASSED\_WITH\_EXCEPTIONS
    - "warnedGates": 0,
    - "passedGates": 1,
    - "failedGates": 0,
    - "passedWithExceptionGates": 0,
    - "eventExecutionTime": "2024-05-24T16:14:48.627942Z"

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* + - **Error Response (e.g., 400 Bad Request):**
    - {
    - "error": {
    - "code": 400,
    - "status": "FAILED",
    - "eventExecutionTime": "2024-05-24T16:14:48.627942Z",
    - "details": [
    - {
    - "field": "csild",
    - "message": "cannot contain letters"
    - }
    - ]
    - }

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* + **2. Fetch results for CI gates (GET /ci/results/{gatrId})**
    - **Successful Response (200 OK) - Example for PASS:**
    - {
    - "metadata": { /\* ... (similar to POST success response) ... \*/ },
    - "findings": [
    - {
    - "remediationTip": "",
    - "failureReason": "",
    - "failureReasonSummary": "",
    - "failureReasonDetail": {},
    - "ruleGroup": {
    - "policy": "Developer Productivity/CVM program. Please refer <link>",
    - "ruleName": "Snyk Critical Vulnerability",
    - "ruleStatus": "PASS",
    - "ruleDescription": "[Soft gate] Source code has upgradable Critical OSS vulnerabilities (cvss score >= 9.0) as identified by Snyk."
    - },
    - "exception": {
    - "expiryDate": "",
    - "exceptionStatus": "",
    - "exceptionId": "",
    - "exceptionApprover": "",
    - "creationDate": ""
    - }
    - }
    - ]

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

*(Other examples provided for FAIL, WARN, PASSED\_WITH\_EXCEPTIONS with varying failureReason, failureReasonSummary, failureReasonDetail, ruleStatus, and exception data).*

* + - **Error Response (e.g., 404 Not Found):**
    - {
    - "error": {
    - "code": 404,
    - "status": "FAILED",
    - "eventExecutionTime": "...",
    - "details": [
    - {
    - "field": "gatrId",
    - "message": "gatrId does not exist"
    - }
    - ]
    - }

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* + **Kafka Contract (for Snyk Scan Results):**
    - Expected fields in build event: csild, project, pipelineName, buildVersion, branch, name (value: "snyk-scan"), siam, exitcode.

**5. LSE CD (Lightspeed Enterprise Continuous Deployment)**

* **Architecture:**
  + A developer initiates deployment.
  + The LSE CD workflow includes a "Process Gates" step which makes a POST /cd/gates call to the Gatr REST API.
  + Interaction with Gatr Components (Gatr REST API, Gatr DB) and LSE components (LSE API, LSE UI, LSE DB) is analogous to LSE CI, but uses CD-specific endpoints and data.
  + LSE UI makes a GET /cd/results/{gatrId} for detailed results.
* **Workflow (LSE CD):**
  + Developer initiates deployment.
  + Pipeline calls /cd/gates endpoint.
  + Gate execution summary is stored in LSE DB.
  + Build status (failed/successful) based on Gatr's overall result.
  + Developers view gate summary in LSE UI.
  + Detailed results via /cd/results endpoint.
* **API Specifications (LSE CD):**
  + **1. Process CD Gates (POST /cd/gates)**
    - **Request Body:**
    - {
    - "csild": 164603,
    - "project": "fin-unify",
    - "pipelineName": "fin-d-cicd-unify-archctr",
    - "buildVersion": "57b-45f4a60-feature-sdlc",
    - "stage": "CD",
    - "environmentName": "gdev",
    - "environmentClass": "dev",
    - "pipelineType": "lse",
    - "manifestName": "helm",
    - "manifestType": "helm", // "helm" for helm manifests
    - "childCsilds": ["175670"], // optional
    - "pipelineExecutionId": "CfCEG\_h65GHEiwAZfDW", // mandatory
    - "branchType": "release" // optional, "release"/"nonRelease", will be mandatory

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* + - **Field Documentation:** Includes CD-specific fields like stage, environmentName, environmentClass, manifestName, manifestType, childCsilds, pipelineExecutionId.
    - **Successful Response:** Similar structure to LSE CI POST /ci/gates success.
    - **Error Response:** Similar structure to LSE CI POST /ci/gates error.
  + **2. Fetch results for CD gates (GET /cd/results/{gatrId})**
    - **Successful Response:** Similar structure to LSE CI GET /ci/results success.
    - **Error Response:** Similar structure to LSE CI GET /ci/results error.

**6. Legacy CI Integration**

* **Architecture:**
  + Similar to LSE CI, but with a "Legacy CI Workflow".
  + "Push Scan to Gatr" step makes a POST /ci/data call to Gatr REST API to upload scan results (e.g., Snyk).
  + "Process Gates" step makes a POST /ci/gates call.
  + A "Gatr UI" can be used to view "CI Gate Results Detail Page" by calling GET /ci/results/{GatrID}.
* **Scope:** Gate results shown in build logs for Legacy Pipelines.
* **API Specifications (Legacy CI):**
  + **1. Upload scan results (POST /ci/data)**
    - **Request Body:**
    - {
    - "csild": 171903,
    - "project": "infra",
    - "pipelineName": "jenkins-r-rhel2",
    - "buildVersion": "jdk11-gradleissue-10-5a20406",
    - "pipelineType": "LSC", // LSC, ICGBUILD, UNIFY
    - "branch": "release/jdk11-test",
    - "branchType": "release", // "release"/"nonRelease", mandatory in future
    - "buildId": "0fffee6f1ad44e", // Loki buildId
    - "buildType": "loki",
    - "buildInstance": "msst",
    - "scanTool": "snyk", // "snyk" or "sonarqube"
    - "scanResultFile": "snykscanresult.json"

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* + - **Successful Response:** { "status": "SUCCESS", "message": "upload complete", "eventExecutionTime": "..." }
    - **Error Response:** { "error": { "code": 400, "status": "FAILED", ... "details": [{"message": "Upload Failed"}] } }
  + **2. Process ci gates (POST /ci/gates)**
    - Request body is similar to LSE CI POST /ci/gates, with pipelineType being "LSC", "ICGBUILD", or "UNIFY". Specific examples provided for Lightspeed Classic, ICG Build, and Unify, mainly differing in pipelineType and example data.
    - Loki Instances (URLs for MSST, ISO, ITS, CPD) are referenced.
  + **3. Fetch results for ci gates (GET /ci/results/{gatrId})**
    - Identical in structure and behavior to LSE CI GET /ci/results. Examples provided for PASS, FAIL, WARN, PASSED\_WITH\_EXCEPTIONS.

**7. Annotations API - VDR (Vulnerability Disclosure Report) Integration**

* **Strategy:**
  + The "Snyk Scan Consumer" in Gatr Components will be enhanced to "Snyk Vuln/VDR/NonSnyk REST API Consumer".
  + Gatr DB will store VDR data alongside other scan results ("Hippo VDR / Others").
* **Note:** No change in the existing CI gates API contract.
* **Notable Points (VDR):**
  + A pipeline will have only one scan result type: either scancxm (Snyk) or VDR.
  + VDR identification: bomFormat: "CycloneDX". If no vulnerabilities, VDR will have an empty vulnerabilities array or it won't exist.
  + Retention period for VDR/scan results in S3 is 14 days.
  + VDR gates are currently only available for CI gating. CD gates integration will occur once a real-time API for CD is available.
  + If VDR scan file is unavailable at any point, CI gates will result in WARN with message "Scan result not available".
  + VDR results will be FAIL if "High/Medium/Critical vulnerabilities" gates are executed and "Scan result not available".
  + The GEM/CSRA/RCE gates will be sunset once all applications move to VDR1.
  + High/Medium/Critical vulnerability gates will be updated for VDR outside HIPPO timelines.
  + Policykit is out of scope.
  + If no tag is available, VDR gates will be skipped.
* **Workflow (VDR):** Same as LSE CI workflow.

**8. VDR Gates (Examples of GET /ci/results/{gatrId} responses)**

* **1. GEM - Hard gate (LSE: Successful Response for overall result FAIL)**
  + ruleGroup.ruleName: "GEMS"
  + ruleGroup.ruleStatus: "FAIL"
  + ruleGroup.ruleDescription: "[Hard gate] This rule will block your pipeline for any fixable GEM (Global Event Management) vulnerabilities."
* **2. RCE - Soft gate (LSE: Successful Response for overall result WARN)**
  + ruleGroup.ruleName: "RCE"
  + ruleGroup.ruleStatus: "WARN"
  + ruleGroup.ruleDescription: "[Soft gate] This rule has identified fixable RCE (Remote code execution) vulnerabilities associated with CSRA (Cyber Security Risk Appetite)."
* **3. CSRA - Soft gate (LSE: Successful Response for overall result WARN)**
  + ruleGroup.ruleName: "CSRA"
  + ruleGroup.ruleStatus: "WARN"
  + ruleGroup.ruleDescription: "[Soft gate] This rule has identified fixable vulnerabilities associated with CSRA (Cyber Security Risk Appetite)."
* *(Legacy examples are also provided with similar content but specific to legacy contexts).*

**9. Parsing Vulnerabilities (from VDR/Scan Data)**

* If a fix is available, it's indicated at two points:
  + affects.versions[] status is UNAFFECTED.
  + analytic.responses[] contains UPDATE. If it only contains NULL/NOT\_FIX, no fix is available.
* Citi specific tagging (GEM, RCE, CSRA) is identified if properties[].name="internal:citi:tags" contains the respective keyword.

**10. Common Schema for findings[].failureDetails**  
The findings array in GET /ci/results and GET /cd/results responses contains a failureDetails array. The structure of elements within failureDetails varies based on the gate type.

* **Threshold Based:**
* {
* "failureType": "THRESHOLD",
* "name": "Green Zone", // or "Reliability Rating"
* "id": "gatr-00654321abcd",
* "type": "TIMEZONE", // or "METRIC\_GREATER\_THAN"
* "datatype": "STRING", // or "NUMBER"
* "required": "1:00 to 5:00", // or "4.0"
* "actual": "2:00", // or "5.0"
* "description": "Timing is in UTC..." // or "Optional Description"

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* **Component Based:**
* {
* "failureType": "COMPONENT",
* "name": "commons-collections:commons-collections",
* "id": "commons-collections:commons-collections",
* "type": "LIBRARY",
* "datatype": "STRING",
* "required": "3.2.2",
* "actual": "3.2.1",
* "description": "Vulnerable Library"

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* **Boolean Based:**
* {
* "failureType": "BOOLEAN",
* "name": "Canary Enablement",
* "id": "canary-enablement",
* "type": "METRIC",
* "datatype": "boolean",
* "required": "true",
* "actual": "false",
* "description": "Optional Description"

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

* **Combined:** Example shows a component-based rule (Snyk) and a gateEvaluationReason ("Snyk Scan Result Not Available") with its own failureDetails (e.g., for OSS Score).
* **Second approach (Not Recommended):** Similar to Combined, but with more generic failure reasons like generalFailureReasons.

**11. CI Gates - Category and Type Details (Summary)**  
This table defines various CI gates. Key columns:

* **Gate Name:** e.g., Snyk High Vulnerability, GEM, RCE, CSRA, Sonar Developer Thresholds, Express Lane Code Quality, ORM Risk Based Retirement.
* **Gate ID:** e.g., gatr-01, gatr-04.
* **Gate Strength:** NON\_ENFORCING, ENFORCING.
* **Service:** CI gates, Tekton controls, CD gates.
* **Policy:** Link to or description of the governing policy.
* **Description:** What the gate checks.
* **Type:** LIBRARY, METRIC, PARAMETER, BRANCH, RISK\_SCORE, ERROR\_CODE, ARTIFACT.
* **Category:** SSTS, CODE\_QUALITY, AUTOMATED\_TESTS, RELEASE, DATA\_QUALITY\_AND\_COMPLIANCE.
* **Sub Category:** HIPPO, NA, EXPRESS\_LANE.
* **GateType:** COMPONENT\_BASED, THRESHOLD\_BASED, BOOLEAN\_BASED.

**12. CD Gates Catalog (Summary)**  
This table defines various CD gates. Key columns:

* **Rule Name:** e.g., CDQR\_hardgate, GEM\_HardGate\_V2, EOVS\_Softgate, SnykGate.
* **Rule Description:** What the rule enforces (BLOCK or WARN).
* **Rule Group:** e.g., CDQR\_RG, TST\_EOVS\_RG.
* **Failure Reason Summary:** Concise explanation of failure.
* **Failure Reason Detail:** More specific details of the failure.
* **Remediation:** How to fix the issue.
* **Policy:** Link to or description of the governing policy.

**13. Roadmap**

* **Current Status (Targeted for June End):**
  + Store Snyk Results
  + Process CD Gates
  + Gates Summary (Share summary of #failed/passed/warned gates, Share overall result)
  + Results Detail (Share details of execution, Share failure reason and remediation)
  + LSE UI Integration (Integrate gating summary, Show detailed findings with actionable feedback)
* **Upcoming Features:**
  + **Q3 2024:**
    - HIPPO: Enable Snyk CI Soft Gate
    - LSE: Snyk CI results on LSE UI
    - LSE: Enable Opt In/Opt out UI
    - Legacy: Enable Opt In/Opt out UI
  + **Q4 2024:**
    - HIPPO: Enable Opt In/Opt out
    - HIPPO: Enable Snyk Hard Gate
    - LSE: Enable Opt In/Opt out UI
    - Legacy: Enable Snyk Hard Gate

**14. Contacts & Feedback**  
Please reach out to:

* @Guledi, Shelly (TECH)
* @Atkinson,William
* @Arroyo, John (TECH)
* @Satpalli, Arunkumar (TECH)  
  ... to provide any feedback.