Audit Ready Framework (HCQ) — Anwendungshandbuch

Version: 0.3.0 · **Stand:** 2026-07-07 · **Sprache:** Deutsch (Primärdokument)
Zielgruppe: Anwender, QM, Entwickler, Auditoren, Projektleitung
Bezug: Web-UI (`frontend/`), API (`/docs`), Traceability `SWREQ-210`

> **Ehrlicher Hinweis:** Das **Audit Ready Framework** (intern: **HCQ**, *Hampa Core Quality*) ist eine **Quality-Intelligence-Plattform** mit API-first-Architektur. Kundenorientiert heißt das Produkt **Audit Ready Framework**; technische Artefakte (Paket `hcq` , URLs, CLI) nutzen weiterhin die Kurzbezeichnung HCQ.
>
> Die Web-Oberfläche zeigt Daten aus dem Backend an; viele Module sind in der UI **primär lesend** (Anzeige, Filter, Detailansicht). Schreiboperationen für die meisten Fachobjekte erfolgen über die REST-API oder zukünftige UI-Erweiterungen. HCQ **ersetzt keine Zertifizierung** — Compliance-Ansichten sind Selbstbewertungen mit dokumentiertem Nachweis, keine Audit-Bescheinigung.
>
> **Schnellstart:** `docs/onboarding/FIRST-10-MINUTES.md` · **Vertriebs-Demo:** `docs/demo/QUICKSTART-DEMO.md` · **Schüler / Einsteiger:** `HCQ-FUER-EINSTEIGER.md` · **Berater / Kunden-API:** `API-ANBINDUNG-KUNDE.md`

Inhaltsverzeichnis

1. Einführung

Was ist HCQ?

Audit Ready Framework (HCQ) ist eine Automotive-fähige Plattform zur Verwaltung und Nachverfolgung von Qualitäts- und Compliance-Artefakten: Softwareanforderungen, Tests, Traceability, Risiken, 8D-Probleme, Änderungen, Releases, APQP/PPAP, IATF-Checklisten und KI-Governance. HCQ verbindet ASPICE-, ISO-26262-, IATF-16949- und ISO-42001-Prozesse in einem einheitlichen UUID-DAG (Traceability-Graph).

Begriff	Bedeutung
Audit Ready Framework	Kundenname / Produktbezeichnung (Website, Lizenz)
HCQ	Technischer Kurzname (Code, CLI `hcq-*`, Paket `hcq`)
Hampa Core Quality	Vollständiger Produktname im README

Die zehn HCQ-Module (Backend)

Die Plattform besteht aus **zehn implementierten Fachmodulen** (Stand: 10/10). Die Web-UI mappt mehrere Querschnittsfunktionen (Audit Cockpit, Compliance-Matrix, KI-Governance, Integrationen) auf dieselben Backend-Pakete.

#	Modul	UI-Route(n)	Handbuch §
1	Requirements Manager	`/requirements`	§5
2	Architecture & Design	`/architecture`	§6
3	DFMEA / Risk Engine	`/risks`	§11
4	8D / Problem Resolution	`/problems`	§10
5	Change & Configuration	`/changes`	§12
6	Test Management	`/tests`	§9
7	QA Dashboard	`/`, `/qa/health`	§3, §4
8	APQP / PPAP Chain	`/apqp`, Traceability Tab Material	§23, §7
9	Traceability Engine	`/traceability`, `/audit-cockpit`	§7, §8
10	IATF 16949 Compliance	`/iatf`	§24

Querschnitt (kein eigenes Modul): Releases/Freigaben (§13), Rollen & RASIC (§14), Benutzerverwaltung (§14.1), Security Events (§15), Norm-Matrix (§16), On-Prem-Lizenz (§17), Digital Twin (§18), CRA (§19), Zertifikatsregister (§20), CRA-Nachweiskette (§21), KI-Governance (§22), Integrationen/Simulation (§2.7).

Vollständige Modul-Landkarte: `docs/README.md` §3.

RBAC-Kurzübersicht

HCQ verwendet **rollenbasierte Zugriffskontrolle (RBAC)** mit Default-Deny (`ADR-0017`). Lesen: alle authentifizierten Rollen. Schreiben/Freigabe: rollenspezifisch.

Rolle	Kurzprofil	Typische UI-Schreibrechte
ADMIN	Vollzugriff	Alles inkl. Benutzerverwaltung (API + UI `/users`)
QM	Qualitätsmanagement	QA-Snapshot, Release-Sign-off, KI-Registry, Security Events
PE	Process Engineer	Architektur anlegen, API-Schreiben auf Engineering-Daten
PM	Projektleitung	QA-Snapshot, APQP, IATF (API), Release-Sign-off
TST	Tester	Testdaten per API
CCB	Change Control Board	Change-Freigaben, Release-Sign-off
SAFETY	Funktionale Sicherheit	DFMEA/HARA per API
DEV	Entwickler	Überwiegend **Lesen**
CUSTOMER	Kunde	Release-Sign-off (Kundenfreigabe)

Vollständige Matrix: `docs/quality/access-control-policy.md`.

Für wen ist HCQ gedacht?

Rolle	Typische Nutzung
QM (Quality Manager)	Dashboard, Audit Cockpit, Probleme, IATF, Compliance, Zertifikatsregister, Digital Twin, CRA, KI-Governance, Security Events
PE (Process Engineer)	Anforderungen, Architektur, Traceability, Tests, Risiken, Changes
DEV (Entwickler)	Lesen: Anforderungen, Architektur, Tests, Traceability
TST (Tester)	Test Management, Abdeckung, Traceability
PM (Projektleitung)	Dashboard, APQP, Releases, RASIC
CCB	Change-Freigaben, Release-Sign-offs
SAFETY	Risikoregister, DFMEA, HARA
CUSTOMER	Release-Sign-offs (Kundenfreigabe)
ADMIN	Vollzugriff inkl. Benutzerverwaltung (API + UI `/users`)

> **Hinweis zu „VIEWER“:** HCQ kennt **keine** Rolle `VIEWER`. Rollen wie `DEV`, `TST` und `CUSTOMER` haben auf viele Module **nur Lesezugriff** — sie verhalten sich faktisch wie Betrachter. Die vollständige Matrix steht in `docs/quality/access-control-policy.md`.

Anmeldung

Umgebung	URL	Hinweis
HCQ Web-UI (Entwicklung/Demo)	`http://localhost:5173`	Fallback-Port **5174**, falls 5173 belegt
Marketing-Website (separates Projekt)	`http://localhost:5174`	`website/` — **nicht** die HCQ-App
API / Swagger	`http://localhost:8000/docs`	Backend-Port 8000
Produktion / Kunde	Installations-URL	Siehe `LIEFERUNG-FIRMSERVER.md`

Simulations-Login (nach `hcq-seed --profile demo` in `ENVIRONMENT=simulation`):

Feld	Wert
Benutzer	`admin`
Passwort	`!!(Fre1a-24)`

> **Ehrlich:** Das Passwort gilt für die **Simulations-DB** (`hcq_simulation.db`), wenn `INITIAL_ADMIN_PASSWORD` vor dem ersten Start gesetzt wurde. `development` und `simulation` sind getrennte Datenbanken. Details: `FIRST-10-MINUTES.md` .

Anmeldung:

1. Browser öffnen → Benutzername und Passwort eingeben → **Anmelden**.
2. Bei erstem Login mit Initialpasswort: Seite **Passwort festlegen** (Pflicht vor App-Zugriff), sofern `INITIAL_ADMIN_MUST_CHANGE_PASSWORD` nicht auf `false` steht.
3. Ist MFA aktiv: 6-stelligen TOTP-Code oder Backup-Code eingeben.

> **On-Prem (SWREQ-231):** HCQ hat **keine** offene Self-Registration. Neue Benutzer werden vom **ADMIN** unter **Governance → Benutzer** per **Einladungslink** angelegt. Der Admin kennt kein Passwort; der Link wird einmal angezeigt und manuell weitergegeben (Teams/USB — air-gap-tauglich, kein E-Mail-Gateway nötig). Der eingeladene Benutzer öffnet `/invite?token=…` , legt sein Passwort fest und wird direkt angemeldet. Konfiguration: `HCQ_PUBLIC_APP_URL` (Basis-URL der Web-UI), `HCQ_INVITE_TOKEN_EXPIRE_HOURS` (Standard 72 h).

Sprache (DE | EN)

Oben rechts in der Seitenleiste: **Sprache** umschalten zwischen **Deutsch** (Standard) und **Englisch**. Die Einstellung wird im Browser (`localStorage` , Schlüssel `hcq-locale`) gespeichert.

Navigation

Die linke Seitenleiste gruppiert Module in fünf Bereiche:

Bereich	Module
Übersicht	QA Dashboard, QA Health
Engineering	Anforderungen, Architektur, Traceability
Qualität	Audit Cockpit, Test Management, 8D/Probleme, DFMEA/Risiko, Change & Config
Governance	Releases, Rollen & RASIC, **Benutzer**, *Security Events*
Compliance	Integrationen (Sim), Norm-Matrix, KI-Governance, APQP/PPAP, IATF 16949

* **Benutzer** nur für **ADMIN**; Security Events nur für **ADMIN** und **QM** sichtbar.

Statusanzeige (Kopfzeile)

- **API online · vX.Y.Z** — Backend erreichbar, Version angezeigt.
- **API offline** — Backend nicht erreichbar; Daten können nicht geladen werden.
- **DAG enthält Zyklen** — Traceability-Graph ist nicht azyklisch (kritisch für Audits).
- **Badge „Simulation" / „Dev"** — nur bei `ENVIRONMENT=simulation` bzw. `development` (in Produktion ausgeblendet).

Simulations-/Entwicklungsmodus (Banner)

Wenn das Backend **nicht** in Produktion läuft, erscheint oberhalb der Kopfzeile ein **amber-farbener Banner**:

Umgebung	Banner (DE)
`simulation`	**SIMULATIONSMODUS — Demodaten, keine Produktivdaten**
`development`	**ENTWICKLUNGSMODUS — Nur für lokale Entwicklung**

Zusätzlich liefert `GET /health` das Feld `"environment"` (z. B. `"simulation"`). In **Produktion** sind Banner, Badge und Simulations-Button **nicht sichtbar**.

---

2. Erste Schritte

> **Onboarding:** Schritt-für-Schritt in `docs/onboarding/FIRST-10-MINUTES.md`
> **5-Minuten-Demo:** `docs/demo/QUICKSTART-DEMO.md`

2.0 Zugangsdaten & Ports (Kurzreferenz)

Was	Wert
HCQ-UI	`http://localhost:5173` (Fallback 5174)
Marketing-Website	`http://localhost:5174` (`website/` , getrennt von HCQ)
API	`http://localhost:8000`
Simulation: Benutzer	`admin`
Simulation: Passwort	`!!(Fre1a-24)`
Demo-Seed	`hcq-seed --profile demo --force` (nur `simulation` / `development`)

2.0a Erklärungsmodus (Learning by doing, `SWREQ-220`)

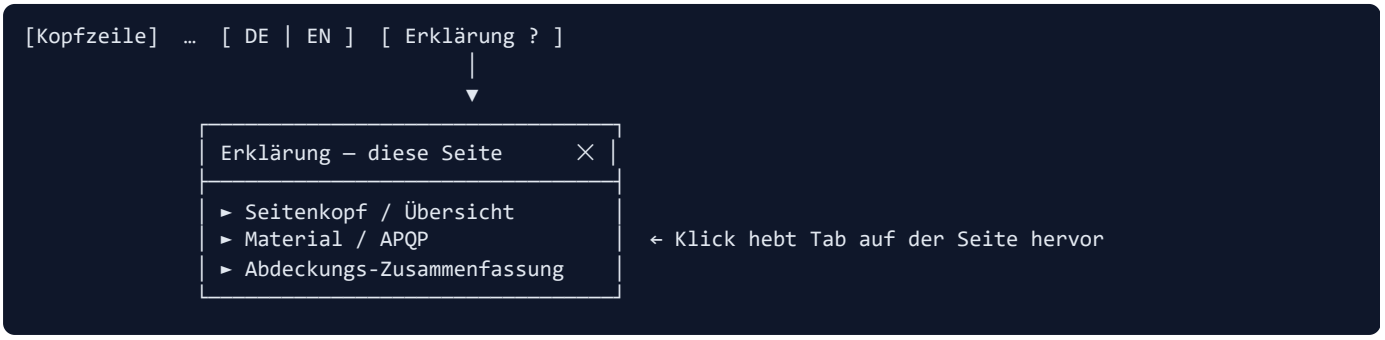
Der **Erklärungsmodus** ist die eingebaute Kontexthilfe der HCQ-Oberfläche: verständliche Texte **direkt am Bildschirm**, ohne PDF durchzusuchen. Gedacht für alle Rollen — nicht nur Qualitätsmanager.

Zielgruppe	Nutzen
Entwickler	Verstehen, welche Felder und Status in Anforderungen, Architektur und Traceability gemeint sind
Projektleiter	Schnell erfassen, was Dashboard-Karten und Freigabe-Indikatoren bedeuten
IT / DevOps	Integrationen, Simulation, Security Events und Profil/Sicherheit erklärt bekommen
QM / Audit	Zusätzlich zum Handbuch: UI-Funktionen und Demo-Grenzen auf einen Blick

Schritt für Schritt

Schritt	Was Sie sehen	Was passiert
1	Nach Login: Kopfzeile oben rechts	Button **„Erklärung“** (EN: **Help**, Icon ?) neben Language Switcher
2	Button **Erklärung** klicken	Side-Panel **„Erklärung — diese Seite“** öffnet sich rechts
3	Liste im Panel durchscrollen	Einträge beziehen sich **nur auf die aktuell geöffnete Seite** (z. B. Traceability)
4	Eintrag im Panel anklicken	Zugehöriges UI-Element auf der Seite wird hervorgehoben (Rahmen um Button, Karte oder Feld)
5	**Esc** oder ✕	Panel schließen; Button **Erklärung** erneut klicken zum Wiederöffnen
6	**DE | EN** umschalten	Erklärungstexte wechseln mit der UI-Sprache

UI-Mock (Beschreibung):



Beispiel: „Was bedeutet Materialnummer in Traceability?“

1. Navigieren Sie zu **Traceability** (Seitenleiste).
2. **Erklärung** in der Kopfzeile aktivieren.
3. Im Panel den Eintrag **„Material / APQP“** wählen — der Tab bzw. das Eingabefeld wird hervorgehoben.
4. Lesen Sie: Materialnummer eingeben → APQP-/PPAP-Kette für dieses Material anzeigen. Leerer Treffer bedeutet: keine Verknüpfung in den Demo-Daten (Seed).
5. **Tipp im Panel:** Probieren Sie eine Materialnummer aus dem Demo-Seed (Integrations-Sync).

So lernen Sie **am lebenden Bildschirm**, was ein Fachbegriff in *dieser* Maske bedeutet — ohne QM-Vorkenntnisse vorauszusetzen.

Aktion	Wirkung
Erklärung klicken	Side-Panel öffnet sich mit Erklärungen **für die aktuelle Seite**
Eintrag im Panel wählen	Zugehöriges UI-Element auf der Seite wird hervorgehoben (`data-help-id`)
Esc oder ✕	Panel schließen
DE | EN	Sprache wie gewohnt über Language Switcher — Erklärungstexte folgen der UI-Sprache

Abgedeckte Bereiche: QA-Dashboard, QA Health, Anforderungen, Architektur, Traceability (inkl. Material/APQP), Audit Cockpit, Tests, 8D, DFMEA, Changes, Releases, Rollen, Security Events, Integrationen, Compliance, KI-Governance, APQP/PPAP, IATF, Profil/Sicherheit.

> **Ehrlicher Hinweis:** Der Erklärungsmodus beschreibt **UI-Funktionen und Demo-Grenzen** — er ersetzt nicht das vollständige `Anwendungshandbuch` (`SWREQ-210`) noch externe Normen oder Audits.

2.1 Login

Schritt	Aktion
1	URL der HCQ-Installation öffnen
2	**Branche / Compliance-Rahmen** wählen: **Automotive** (Standard) oder **Robotik** — siehe §2.1a
3	Benutzername + Passwort eingeben (öffentliche Demo: zuerst Branche wählen, dann **Demo starten**)
4	Optional: **Mit Passkey anmelden** (wenn WebAuthn aktiviert und Passkey registriert)
5	Bei MFA: Authentifizierungscode eingeben

Die Branchen-Voreinstellung wird **lokal im Browser** gespeichert (`localStorage` , Schlüssel `hcq.complianceDomain`) und steuert u. a. die Compliance-/Norm-Matrix sowie Diagnose-Profile (Automotive = IATF/ASPICE/ISO 26262; Robotik = Roadmap-Scaffold ISO 10218, ISO/TS 15066, … — `SWREQ-229`).

Häufige Fehler: „Anmeldung fehlgeschlagen" — Zugangsdaten prüfen; bei Rate-Limit kurz warten. Bei `must_change_password` erscheint kein Dashboard, sondern die Passwort-Seite.

2.1a Branchenwahl Automotive / Robotik (`SWREQ-229`)

HCQ unterscheidet zwei **Compliance-Rahmen** für Norm-Matrix, Audit-Schwerpunkte und Diagnose-Fragebögen. Die Wahl ist **kein Login-Account** und **keine Zertifizierung** — sie filtert nur, welche Normen und UI-Texte angezeigt werden.

Element	Ort	Verhalten
Umschalter	Login-Seite (über Anmeldebutton bzw. **Demo starten**)	Zwei Buttons: **Automotive** / **Robotik**
Umschalter	Kopfzeile (rechts, neben Simulation/Sprache)	Jederzeit wechseln — wirkt sofort auf Compliance & Diagnose
Speicherung	Browser `localStorage`	Bleibt über Sitzungen erhalten (pro Browser/Profil)
Backend-Filter	`GET /compliance/*?domain=automotive|robotics`	Matrix und Export folgen der gewählten Branche

Automotive (Standard): IATF 16949, ISO 26262, VDA 6.3, ASPICE v4.0 — kuratiertes Mapping mit ehrlichem Status (`fulfilled` / `partial` / `roadmap`).

Robotik (Roadmap): ISO 10218, ISO/TS 15066, ISO 13849, IEC 62443 — **Scaffold**, kein vollständiges Klausel-Mapping wie Automotive. Die UI zeigt einen **Roadmap-Hinweis** (Selbstbewertung, nicht zertifiziert).

Öffentliche Demo (`VITE_DEMO_AUTO_LOGIN`): Die Demo **überspringt nicht mehr** die Login-Karte — zuerst Branche wählen, dann **Demo starten** (ein Klick, kein Passwort für Gast `demo`). Der Umschalter bleibt nach dem Login in der Kopfzeile sichtbar.

Qualifikation: `tests/test_compliance_domain_ui.py` , `tests/test_compliance_matrix.py` (Robotik-Domain), Frontend `frontend/src/test/compliance-domain.test.ts` .

2.2 Passwort ändern und Einladung annehmen

Passwort ändern (angemeldeter Benutzer)

Profil & Sicherheit (Klick auf Benutzername oben in der Seitenleiste):

Feld	Bedeutung	Pflicht?
Aktuelles Passwort	Bestätigung der Identität	Ja
Neues Passwort	Mindestens 8 Zeichen	Ja
Neues Passwort bestätigen	Muss übereinstimmen	Ja

Einladung annehmen (On-Prem, `SWREQ-231`)

Neue Benutzer erhalten vom Administrator einen **Einmal-Link** (`/invite?token=…`), kein vorgegebenes Passwort.

Schritt	Aktion
1	Einladungslink öffnen (Teams/USB — kein E-Mail-Versand in HCQ)
2	Anzeigename und Benutzername werden angezeigt (Token-Validierung)
3	Persönliches Passwort festlegen (min. 8 Zeichen)
4	**Passwort setzen & anmelden** — direkter App-Zugang per JWT

Abgelaufene oder bereits verwendete Links werden abgewiesen. Details: §14.1.

2.3 Profil

Anzeige	Bedeutung
Benutzername	Eindeutige Anmelde-ID
Vollständiger Name	Anzeigename
Rolle	Ihre HCQ-Rolle (z. B. `QM` , `DEV`) — steuert Berechtigungen
MFA-Status	Aktiv / Deaktiviert

2.4 MFA (optional, TOTP)

1. **MFA aktivieren** klicken.
2. Secret in Authenticator-App scannen (z. B. Google Authenticator, Aegis).
3. 6-stelligen Code eingeben → **Bestätigen & aktivieren**.
4. **Backup-Codes sofort sichern** — sie werden nur einmal angezeigt.

Deaktivierung: Passwort + TOTP/Backup-Code erforderlich.

2.5 Passkey / WebAuthn (optional)

Zusätzliche Anmeldemethode; Passwort und TOTP bleiben primär.

1. **Passkey registrieren** → Browser-Dialog bestätigen (Fingerabdruck, PIN, Security Key).
2. Am Login: **Mit Passkey anmelden** (Benutzername muss vorher eingetragen sein).

> Passkeys sind **optional** und abhängig von Server-Konfiguration (`WEBAUTHN_ENABLED`).

2.6 Demo-Modus (Simulation)

Für Vertriebs- und Schulungsdemos steht ein **Simulationsprofil** bereit (`ADR-0031`). Es verwendet **keine echten** SAP-, DOORS- oder PLM-Systeme, sondern **kuratierte CSV-Exporte** in `data/demo/` (konfigurierbar via `DEMO_DATA_DIR`) mit realistischen Automotive-Feldnamen und IDs (`DOORS-REQ-xxx` , `SAP-CHG-xxx` , `PLM-PART-xxx`).

Schritt	Aktion
1	`ENVIRONMENT=simulation` setzen
2	`INITIAL_ADMIN_PASSWORD=!!(Fre1a-24)` und optional `INITIAL_ADMIN_MUST_CHANGE_PASSWORD=false`
3	`hcq-seed --profile demo --force` ausführen (lädt CSV-Exporte)
4	Backend und Frontend starten (siehe `FIRST-10-MINUTES.md`)
5	Login: `admin` / `!!(Fre1a-24)` — oder öffentliche Demo: **Branche wählen** → **Demo starten**
6	UI: **Banner „SIMULATIONSMODUS“**, Badge **Simulation**, Button **Simulation** in der Kopfzeile
7	**Automotive / Robotik** in der Kopfzeile umschalten → **Compliance** / **Norm-Matrix** prüfen
8	Button **Simulation** oder Seite **Integrationen (Sim)** → **Synchronisieren**
9	**Anforderungen** öffnen — Badge `DOORS-REQ-001` bei synchronisierten Einträgen

API-Nachweis: `GET /integrations/status` , `GET /integrations/doors/requirements` , `GET /integrations/plm/parts` , `GET /demo/drawings/{filename}` , `POST /integrations/sync-all` (nur Simulation; in Produktion `403`).

> **Ehrlich:** Die CSV-Dateien sind Demo-Artefakte, keine echten Kundendaten.

2.7 Integrationen (Simulation)

Route: `/integrations` · **Sichtbar:** alle Rollen · **Sync:** nur `simulation` / `development`

Zweck

Übersicht der **simulierten Fremdsystem-Exporte** (DOORS, SAP ECM/MM, PLM) und manueller Re-Import per UI-Button — ohne echte Systemanbindung.

Oberfläche

Element	Bedeutung
Synchronisieren	`POST /integrations/sync-all` — CSV aus `data/demo/` einlesen
Umgebung	`simulation` / `development` / `production`
DOORS-Vorschau	Erste Zeilen aus `doors_requirements_export.csv`
SAP-Änderungen	`sap_change_masters.csv`
SAP-Materialien	`sap_materials.csv` (Materialnummern für Traceability Tab Material)
PLM-Teile	`plm_parts.csv` inkl. Zeichnungslink
Letzter Sync	Zeitstempel, gelesene Datensätze

Der Kopfzeilen-Button **Simulation** führt Sync aus und navigiert hierher.

Typischer Workflow

1. Nach frischem Seed oder DB-Reset: **Synchronisieren** klicken.
2. Vorschau prüfen — Dateipfade und Datensatzanzahl.
3. **Anforderungen** / **Architektur** / **Change & Config** auf importierte IDs prüfen.
4. **Traceability** → **Material / APQP** mit Demo-Materialnummer testen (siehe §7).

Tipps

- In **Produktion** liefert Sync `403` — beabsichtigter Safety-Guard.
- Leere Vorschau: `hcq-seed --profile demo --force` oder Sync erneut ausführen.

3. QA Dashboard

Route: `/` · **Prozess:** MAN.6 · SUP.1

Zweck

Konsolidierte **Qualitätslage auf einen Blick**: Anforderungen, Testabdeckung, Bestehensquote, offene Probleme, Risiken und Traceability-Gesundheit.

Wer darf was?

Operation	Rollen
Ansehen	Alle authentifizierten Rollen
Metrik-Snapshot erstellen	QM, PM, ADMIN

Oberfläche

- **KPI-Karten** oben: Anforderungen (verifiziert), Abdeckung %, Pass-Rate, offene Probleme.
- **Diagramme:** Statusverteilung, Testabdeckung (Pie), Probleme nach Schweregrad, ASIL/DFMEA, Qualitätstrend (Snapshots).
- **Aktualisieren** — Daten neu laden.
- **Snapshot** — aktuellen KPI-Stand persistieren (nur berechtigte Rollen).

Datenfelder (KPI-Übersicht)

Feld	Bedeutung	Beispiel	Pflicht?
`requirements.total`	Anzahl SW-Anforderungen	`142`	—
`requirements.verified`	Status `verified`	`98`	—
`tests.coverage_ratio`	Anteil getrackter Anforderungen mit Test	`0.87`	—
`tests.latest_pass_rate`	Bestehensquote letzter Lauf	`0.95`	—
`problems.open`	Offene Problemberichte	`3`	—
`traceability_health.orphan_count`	Anforderungen ohne Test	`5`	—
`risks.high_risk_rating_count`	DFMEA mit Bewertung **H**	`2`	—

Typischer Workflow

1. Dashboard öffnen → Lage prüfen.
2. Rote/warnende KPIs identifizieren (kritische Probleme, niedrige Abdeckung).
3. In Detailmodul springen (z. B. Traceability bei Orphans).
4. Optional: **Snapshot** für Trend-Dokumentation (MAN.6).

Häufige Fehler / Tipps

- **Pass-Rate leer:** Noch kein Testlauf erfasst — normal bei Neuinstallation.
- **DAG zyklisch:** Trace-Links prüfen; Zyklen blockieren audit-sichere RTM.
- Snapshot-Button ausgegraut: Ihre Rolle darf keine QA-Metriken schreiben.

4. QA Health

Route: `/qa/health` · **Prozess:** SUP.1 · MAN.6

Zweck

Detailansicht zu **Traceability-Lücken**, **Audit-Aktivität** und **Metrik-Snapshots** — ergänzt das Dashboard um operative Gesundheitsindikatoren.

Wer darf was?

Lesen: alle Rollen. Snapshots: siehe QA Dashboard (QM, PM, ADMIN).

Oberfläche

- **Abdeckungslücken** — Liste verwaister Anforderungs-IDs (`orphan_requirements`).
- **Zusammenfassung** — Abdeckung %, hängende Knoten, DAG-Status.
- **Audit-Aktivität** — Aufschlüsselung nach Aktion, Entitätstyp, Akteur.
- **Metrik-Snapshots** — Tabelle historischer KPI-Stände.

Datenfelder

Feld	Bedeutung	Beispiel	Pflicht?
`orphan_requirements`	SWREQ ohne verifizierenden Test	`["SWREQ-042"]`	—
`dangling_node_count`	Knoten ohne gültige Kante	`0`	—
`dag_acyclic`	Graph ohne Zyklen	`true`	—
`audit_summary.by_action`	Aktionen (create, update, …)	`{create: 120}`	—
`snapshot_id`	Snapshot-Kennung	`QA-METRIC-003`	Ja (bei Snapshot)
`kpis.*`	KPI-Schnappschuss	siehe Dashboard	—

Typischer Workflow

1. QA Health öffnen nach Release-Vorbereitung.
2. Orphan-Liste exportieren mental → Testfälle nachziehen.
3. Audit-Aktivität auf ungewöhnliche Muster prüfen.

Tipps

- „Keine Abdeckungslücken“ = alle Anforderungen haben mindestens einen verifizierenden Testfall.
- Bei `DAG ZYKLISCH` sofort Traceability-Team einschalten.

---

5. Anforderungen (Requirements)

Route: `/requirements` · **Prozess:** SWE.1

Zweck

Verwaltung und Einsicht in **Softwareanforderungen** (`SWREQ-*`) inkl. Lebenszyklus, Priorität, Verifikationsmethode und Trace-Links.

Wer darf was?

Operation	Rollen
Lesen	Alle
Anlegen/Ändern (API)	PE, PM, ADMIN
UI-Schreiben	**Nein** (nur Lesen in aktueller UI)

Oberfläche

- **Filter:** Kategorie, Status, Priorität.
- **Tabelle:** ID, Titel, Kategorie, Priorität, Status, Aktualisiert.
- **Zeile klicken** → Detail-Dialog mit Beschreibung und Trace-Tabs (Upstream/Downstream).
- **Paginierung** — 20 Einträge pro Seite.
- **Aktualisieren** — Daten neu laden.

Datenfelder-Tabelle

Feldname	Bedeutung	Beispiel	Pflicht?
`req_id`	Menschenlesbare ID	`SWREQ-010`	Ja
`uuid`	Interne UUID (DAG-Knoten)	`a1b2c3d4-…`	Ja (auto)
`title`	Kurztitel	`JWT-Authentifizierung`	Ja
`description`	Volltext der Anforderung	`Das System SHALL …`	Ja
`category`	Anforderungstyp	`FUNC`, `PERF`, `IF`, `CON`, `DATA`	Ja
`priority`	Priorität	`M` (Muss), `S` (Soll), `K` (Kann)	Ja
`source`	Herkunft/Stakeholder	`SYS-006`, `Kunde XY`	Ja
`verification_method`	Verifikationsart	`T` Test, `A` Analyse, `R` Review, `I` Inspektion, `D` Demo	Ja
`status`	Lebenszyklus	`draft` → `verified` → `deprecated`	Ja
`created_at` / `updated_at`	Zeitstempel (UTC)	`2026-06-01T10:00:00Z`	Auto

Kategorien erklärt:

Code	Bedeutung
FUNC	Funktionale Anforderung
PERF	Performance
IF	Schnittstelle
CON	Constraint / Einschränkung
DATA	Datenanforderung

Status-Lebenszyklus:

`draft` → `review` → `approved` → `implemented` → `verified` → (`deprecated`)

Typischer Workflow

1. Filter auf `status=review` setzen → Review-Kandidaten finden.
2. Anforderung öffnen → Upstream (SYS) und Downstream (Tests, Design) prüfen.
3. Fehlende Links über API/Traceability ergänzen (UI: nur Anzeige).
4. Nach Verifikation Status auf `verified` setzen (API).

Häufige Fehler / Tipps

- **Keine Downstream-Links:** Anforderung ist nicht implementiert/getestet — Audit-Risiko.
- `priority=M` ohne Test: Widerspruch — Muss-Anforderungen brauchen Verifikation.
- UUID im Dialog ist für API/Dossier-Suche relevant, nicht für Alltagsarbeit.

6. Architektur & Design

Route: `/architecture` · **Prozess:** SWE.2 · SWE.3

Zweck

Darstellung der **Softwarearchitektur** (Komponenten, Schnittstellen) und **Detaildesign-Einheiten** gemäß SWE.2/SWE.3.

Wer darf was?

Operation	Rollen
Lesen	Alle
Anlegen (UI)	PE, ADMIN

Oberfläche

Tabs: Komponenten | Schnittstellen | Detaildesign

- **Komponente hinzufügen / Schnittstelle hinzufügen / Design-Einheit hinzufügen** (nur PE/ADMIN).
- Tabellen mit Status-Badges, Aktualisieren-Button.
- Klick auf eine **Komponente** oeffnet Details inkl. Bauteilname und **Zeichnung oeffnen** (wenn PLM-Sync Zeichnungsreferenz mitgeliefert hat).

Datenfelder — Komponenten (`ARC-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`arc_id`	Komponenten-ID	`ARC-API`	Ja (auto)
`name`	Name	`API Gateway`	Ja
`description`	Beschreibung	`FastAPI REST-Schicht`	Ja
`component_type`	Typ	`subsystem` , `component` , `module` , `interface_layer`	Ja
`responsibility`	Verantwortungsbereich	`Authentifizierung`	Ja
`status`	Designstatus	`draft` , `reviewed` , `approved`	Ja
`part_name`	Bauteilname (PLM-Sync)	`Gehäuse Aluminium VACE`	Nein
`drawing_url`	Link zur Zeichnung	`/demo/drawings/PLM-PART-002-rev-B.pdf`	Nein
`drawing_container`	Dokumentencontainer	`data/demo/drawings/...`	Nein

Datenfelder — Schnittstellen (`IF-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`if_id`	Schnittstellen-ID	`IF-001`	Ja (auto)
`name`	Name	`Requirements API`	Ja
`provider_component`	Anbieter (`arc_id`)	`ARC-API`	Ja
`consumer_component`	Konsument	`ARC-FE`	Ja
`direction`	Richtung	`provided` , `required` , `bidirectional`	Ja
`contract`	Vertrag/Beschreibung	`OpenAPI 3.1`	Ja
`protocol`	Protokoll	`REST` , `CAN`	Ja
`status`	Designstatus	`approved`	Ja

Datenfelder — Detaildesign (`DD-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`dd_id`	Design-Einheit-ID	`DD-AUTH-01`	Ja (auto)
`name`	Name	`TokenService`	Ja
`parent_component`	Übergeordnete Komponente	`ARC-AUTH`	Ja
`design_details`	Detailbeschreibung	Pseudocode, Algorithmen	Ja
`status`	Designstatus	`reviewed`	Ja

Typischer Workflow

1. Komponentenbaum aufbauen (Subsystem → Module).
2. Schnittstellen zwischen Komponenten definieren.
3. Detaildesign-Einheiten zu Komponenten verfeinern.
4. Status auf `approved` setzen vor Release.

Tipps

- Ohne Schreibberechtigung: Button „Nur-Lese-Rolle" — Daten per API pflegen oder PE bitten.
- `provider` / `consumer` müssen existierende `arc_id` referenzieren.

---

7. Traceability

Route: `/traceability` · **Prozess:** SWE.6 · RTM

Zweck

Requirements Traceability Matrix (RTM) und **Trace Explorer** — bidirektionale Verknüpfung über den UUID-DAG (`derives_from` , `implements` , `verifies` , `changes`).

Wer darf was?

Lesen: alle. Links anlegen/ändern: PE, PM, ADMIN (API).

Oberfläche

Tab RTM: Matrix mit Spalten Abgeleitet von | Anforderung | Implementiert durch | Verifiziert durch.

Tab Trace Explorer: Anforderung wählen → grafische Kette Upstream → Anforderung → Downstream.

Tab Material / APQP (`SWREQ-096`): Materialnummer eingeben → verknüpfte APQP-Phasen, PPAP-Elemente, Nachweise und DAG-Links (Anforderungen, Tests).

Tab Material / APQP — Felder

Feld / Spalte	Bedeutung	Beispiel
Eingabe Materialnummer	SAP- oder PLM-Nummer	`100000002` , `MAT-002`
`resolved_material_number`	Aufgelöste Nummer nach Demo-Mapping	`100000002`
Badge **Demo-Simulation**	SAP↔PLM-Mapping nur in Simulation	sichtbar bei Demo-Daten
`plm_part`	PLM-Teil (ID, Revision, Name)	`PLM-PART-002`
Spalte **Kategorie**	Dokumenttyp	`APQP` , `PPAP` , `Design`
Spalte **Phase**	APQP-Phase	`APQP-P3`
Spalte **Dokument**	Lieferobjekt / Elementname	`DFMEA`
Spalte **Status**	Bearbeitungsstatus	`complete`
Spalte **Nachweis**	Evidence-Referenz	`SWREQ-080` , Test-ID
Verknüpfte Anforderungen	DAG-Upstream/Downstream	`SWREQ-*` Chips
Verknüpfte Tests	Verifizierende Testfälle	`TC-*` Chips

API: `GET /traceability/by-material/{materialNr}`

Demo-Mapping (Simulation): SAP `MAT-002` ↔ PLM-Material `100000002` (ECU-Leiterplatte).

Typischer Workflow — RTM & Explorer

1. RTM öffnen → Abdeckungslücken in Spalte „Verifiziert durch“ spotten.
2. Trace Explorer: konkrete SWREQ auswählen → Kette visualisieren.
3. Fehlende Links in API ergänzen.

Typischer Workflow — Materialnummer (Auditor)

1. Tab **Material / APQP** öffnen.
2. Materialnummer eingeben (Demo: `100000002` oder `MAT-002`) → **Suchen**.
3. Aufgelöste Nummer und PLM-Teil prüfen; Badge **Demo-Simulation** beachten.
4. Tabelle **Dokumente** durchgehen (APQP-Phasen, PPAP-Elemente, Nachweise).
5. Verknüpfte Anforderungen/Tests als DAG-Ergänzung nutzen.
6. Optional: gleiche Nummer im **Audit Cockpit** suchen (siehe §8.1).

Tipps

- Leere Spalte „Verifiziert durch“ in RTM = keine Testverknüpfung (Orphan-Risiko).
- `kind` am Knoten zeigt Artefakttyp (`requirement` , `test_case` , `component` , …).
- **404 / nicht gefunden:** Material nicht im Demo-Stamm — Seed/Sync prüfen.
- In Produktion ohne SAP/PLM-Anbindung: Feature liefert nur echte Daten, kein Demo-Mapping.

Datenfelder — RTM-Eintrag

Feldname	Bedeutung	Beispiel	Pflicht?
`requirement.ref_id`	Anforderungs-ID	`SWREQ-010`	Ja
`derives_from`	Upstream-Knoten	SYS-Anforderungen	—
`implemented_by`	Design/Code-Knoten	`ARC-API`, `DD-*`	—
`verified_by`	Test-Knoten	`TC-042`	—
`link_type`	Kantentyp	`verifies`	Ja (bei Link)

---

8. Audit Cockpit

Route: `/audit-cockpit` · **Prozess:** SUP.10 · IATF

Zweck

Read-only Traceability-Dossier für Auditoren: Ein Artefakt suchen → vollständige Rückverfolgbarkeit inkl. Lückenanalyse (Gap Flags).

Wer darf was?

Lesen: alle. **Keine Schreibfunktion** in der UI.

Oberfläche

- **Suchfeld** — ID oder Stichwort (z. B. `SWREQ-001`, `login`).
- **Ergebnisliste** — Treffer mit `kind`, Status, Traceable-Hinweis.
- **Dossier-Ansicht:**
  - Vollständigkeits-Score (%), Ampel-Status (grün/gelb/rot).
  - Trace-Kette (Upstream ← Artefakt → Downstream).
  - Sektionen: Anforderungen, Design, Tests, Risiken, Probleme, Changes, Baselines, Freigaben.
  - **Gap-Analyse** — was für ein sauberes Audit fehlt.
- **JSON exportieren** — Dossier herunterladen.

Datenfelder — Dossier (Auswahl)

Feldname	Bedeutung	Beispiel	Pflicht?
`completeness_score`	Vollständigkeit 0–100	`78`	—
`overall_status`	Ampel	`green`, `amber`, `red`	—
`gaps[].code`	Lücken-Code	`NO_TEST_COVERAGE`	—
`gaps[].label`	Lesbare Beschreibung	`Keine Testabdeckung`	—
`tests[].latest_status`	Letzter Teststatus	`pass`, `fail`	—
`risks[].dfmea_risk_rating`	DFMEA L/M/H	`high` → **H**	—
`approvals[].decision`	Freigabeentscheidung	`approve`	—

Typischer Workflow

1. Auditor nennt Referenz (z. B. `SWREQ-060`).
2. Suchen → Dossier öffnen.

3. Gap-Analyse durchgehen → offene Punkte dokumentieren.
4. JSON exportieren als Audit-Anhang.

Tipps

- PDF-Export ist **Roadmap** — aktuell JSON.
- Rote Gaps sind Blocker für Release-Freigabe, gelb sind Warnungen.

8.1 Typische Auditor-Workflows

A — Software-Anforderung (SWREQ)

Schritt	Modul	Aktion
1	Audit Cockpit	`SWREQ-060` suchen → Dossier öffnen
2	Dossier	Vollständigkeits-Score und Gap-Analyse lesen
3	Traceability	Tab Explorer → gleiche SWREQ → Kette prüfen
4	Tests	Tab Abdeckung → verifizierende TCs
5	Export	JSON-Dossier als Audit-Anhang

B — Release-Freigabe mit Zeitstempel

Schritt	Modul	Aktion
1	Releases	Release `REL-*` öffnen
2	Readiness	Pflichtkriterien + fehlende Rollen prüfen
3	Sign-offs	Liste mit `decided_at` (UTC, serverseitig) dokumentieren
4	Audit Cockpit	Release-ID suchen → Freigaben-Sektion im Dossier

C — Materialnummer → APQP/PPAP (IATF 8.3 / Lieferantenaudit)

Schritt	Modul	Aktion
1	Integrationen (Sim)	Sync-Status / SAP-Materialien prüfen (Demo)
2	Traceability	Tab **Material / APQP** → `100000002` suchen
3	Ergebnis	PLM-Teil, Dokumententabelle, Nachweise (`evidence_ref`)
4	APQP / PPAP	Phasen- und PPAP-Detail zum Teil öffnen
5	Audit Cockpit	`PPAP-*` oder `PLM-PART-*` als Stichprobe

> **Demo-Hinweis:** Workflow C nutzt Simulationsdaten. In Produktion sind echte SAP/PLM-Schnittstellen erforderlich — derzeit Roadmap für Live-Connector.

D — IATF-Readiness

Schritt	Modul	Aktion
1	IATF 16949	`audit_ready` und Heatmap prüfen
2	Compliance	Tab IATF → Klausel-Nachweise
3	QA Health	Orphans und DAG-Status
4	Framework Self-Audit	`HCQ-SELF-AUDIT-PACKAGE.md`

8.2 Audit-Vorbereitung (ASPICE · TISAX · IATF)

Route: `/audit-preparation` · **API:** `POST /audit-readiness/analyze` · **Anforderungen:** SWREQ-221/222/223 · **Design:** ADR-0035

Zweck

Regelbasierte **Gap-Analyse** vor externen Assessments. Drei Audit-Profile:

Audit-Typ (`audit_type`)	Fokus	Disclaimer (Kern)
`aspice_cl2`	ASPICE v4.0 CL2 — SYS.2, SWE.1–6, SUP.1/8–10, MAN.5	Kein CL-Urteil
`tisax`	VDA ISA — RBAC, MFA, Audit-Trail, Security Events, TLS/Krypto-Doku	Kein ENX-Label; Org-Kontrollen als Roadmap
`iatf_16949`	IATF QMS — Trace, APQP/PPAP, 8D, Change, Checkliste, KPI, NCR	Kein IATF-Zertifikat; 8.5 Produktion n/a

Datenquellen: HCQ-Laufzeitdaten (DB) + Repository-Nachweise (`traceability.yaml` , TISAX-Paket, IATF-Landkarte).

Oberfläche

1. **Audit-Typ** — Dropdown: ASPICE CL2 | TISAX | IATF 16949.
2. **Scope (optional)** — Projekt/Materialnummer (Filter: Roadmap).
3. **Analyse starten** — ruft `POST /audit-readiness/analyze` auf.
4. **Ergebnis** — Gesamtscore (%), Ampel, profil-spezifischer Disclaimer, Gap-Tabelle.
5. **Beheben** — Deep-Link zum passenden Modul.

Checks je Profil (Auszug)

ASPICE CL2: STK→SYS-Trace, Testabdeckung, RaC-verifies, ADR/ARC-Links, DAG, Testergebnisse, CR-Genehmigung, 8D, Release-Sign-offs, Baselines, DFMEA, Self-Audit-Artefakte.

TISAX: RBAC/Default-Deny, Access-Control-Policy, MFA, TLS/Krypto-ADRs, Audit-Trail, Security Events/SIEM, Secure SDLC (CI/Gates), Change Control, Incident/8D, TISAX-Readiness-Paket; **ENX** und **physische Sicherheit** immer `partial` (Org-Ebene).

IATF 16949: DAG, Testabdeckung, APQP-Phasen, PPAP (inkl. Kundenfeld), 8D/CAPA, Change Control, IATF-Checkliste, KPI on-target, Major-NCRs, Test-Pass/Fail, IATF-Doku-Landkarte.

Typischer Workflow

1. Audit-Typ wählen → **Analyse starten**.
2. `missing` / `partial` -Befunde priorisieren → **Beheben** → Modul aktualisieren.
3. Analyse wiederholen; TISAX: Org-Lücken (ENX/Physisch) separat planen, nicht als Software-Gap erwarten.

API

```
POST /audit-readiness/analyze
Content-Type: application/json

{"audit_type": "aspice_cl2", "scope": null}
```

Antwort: `overall_score` , `overall_status` , `disclaimer` , `findings[]` mit `process_area` , `status` , `evidence_ref` , `deep_link_hint` .

Route: `/tests` · **Prozess:** SWE.4 · SWE.6

Zweck

Testspezifikationen, **Testfälle**, **Testläufe** und **Anforderungsabdeckung** verwalten und einsehen.

Wer darf was?

Operation	Rollen
Lesen	Alle
Schreiben (API)	TST, PE, ADMIN
UI-Schreiben	**Nein** (Lesen)

Oberfläche

Tabs: Spezifikationen | Testläufe | Abdeckung

- Filter **Stufe** (unit / integration / qualification) bei Spezifikationen.
- Testfall-Zeile klicken → Detail (Schritte, erwartetes Ergebnis).
- Abdeckungstabelle: SWREQ ↔ abgedeckt ja/nein ↔ verifizierende Testfälle.

Datenfelder — Testspezifikation (`TSPEC-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`spec_id`	Spezifikations-ID	`TSPEC-UNIT-01`	Ja
`title`	Titel	`Auth Unit Tests`	Ja
`level`	Teststufe	`unit`	Ja
`source_swreq`	Bezugsanforderung	`SWREQ-150`	Nein
`description`	Beschreibung	Scope der Spec	Ja

Datenfelder — Testfall (`TC-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`tc_id`	Testfall-ID	`TC-001`	Ja
`tspec_id`	Übergeordnete Spec	`TSPEC-UNIT-01`	Ja
`title`	Titel	`Login success`	Ja
`preconditions`	Vorbedingungen	`User exists`	Ja
`steps`	Testschritte (Liste)	`["POST /auth/login", …]`	Ja
`expected_result`	Soll-Ergebnis	`200 + token`	Ja
`priority`	Priorität	`high` , `medium` , `low`	Ja

Datenfelder — Testlauf (`TR-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`run_id`	Lauf-ID	`TR-2026-06-17`	Ja
`executed_at`	Ausführungszeitpunkt	ISO-8601	Ja
`executor`	Ausführender	`ci-pipeline`	Ja
`environment`	Umgebung	`sim` , `prod`	Ja
`passed` / `failed` / `blocked` / `skipped`	Ergebniszähler	Report	—

Datenfelder — Testergebnis

Feldname	Bedeutung	Beispiel	Pflicht?
`status`	Ergebnis	`pass` , `fail` , `blocked` , `skipped`	Ja
`actual_result`	Ist-Ergebnis	`200 OK`	Ja
`notes`	Anmerkungen	Fehlerdetails	Nein

Typischer Workflow

1. Tab Abdeckung → nicht abgedeckte SWREQ identifizieren.
2. Testfälle in Spec-Tab prüfen.
3. Testläufe → Pass-Rate für Release bewerten.
4. Ergebnisse per API/CI einspielen.

Tipps

- `blocked` zählt nicht als Pass — Release-Gates beachten.
- `source_swreq` verknüpft Spec direkt mit Anforderung.

---

10. 8D / Probleme

Route: `/problems` · **Prozess:** SUP.9

Zweck

Problemberichte, **8D-Workflow** (D1–D8), Ursachenanalyse und Korrekturmaßnahmen.

Wer darf was?

Operation	Rollen
Lesen	Alle
Schreiben (API)	QM, PE, ADMIN
UI-Schreiben	**Nein** (Lesen)

Oberfläche

- Statistik-Karten (offen/geschlossen, Schweregrad-Chart).
- **Tab Problemberichte** — Filter Status/Schweregrad, Detail-Dialog mit 8D, RCA, CAs.

Datenfelder — Problembericht (`PR-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`pr_id`	Problem-ID	`PR-001`	Ja
`title`	Kurztitel	`Login timeout in prod`	Ja
`description`	Beschreibung	Ausführlicher Text	Ja
`severity`	Schweregrad	`critical` , `major` , `minor`	Ja
`status`	8D-Status	`open` … `closed`	Ja
`reporter`	Melder	`max.mustermann`	Ja
`affected_items`	Betroffene Referenzen	`["SWREQ-150"]`	Nein
`linked_change_request`	Verknüpfter CR	`CR-005`	Nein
`detected_at`	Erkennungszeitpunkt	ISO-8601	Ja

Datenfelder — 8D-Bericht (D1–D8)

Feld	Bedeutung
`d1_team`	D1 · Team
`d2_problem_description`	D2 · Problembeschreibung
`d3_containment`	D3 · Sofortmaßnahmen / Eindämmung
`d4_root_cause`	D4 · Ursache
`d5_corrective_actions`	D5 · Korrekturmaßnahmen
`d6_implement_verify`	D6 · Umsetzung & Verifikation
`d7_prevent_recurrence`	D7 · Wiederholung verhindern
`d8_closure`	D8 · Abschluss

Datenfelder — Korrekturmaßnahme (`CA-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`assignee`	Verantwortlicher	`pe.user`	Ja
`due_date`	Fälligkeit	`2026-07-01`	Nein
`status`	Status	`open` … `verified`	Ja
`verification_result`	Verifikationsergebnis	`Test TC-099 pass`	Nein

Typischer Workflow

1. Problem melden (API) mit Schweregrad.
2. D3 Containment → D4 RCA (5-Why/Ishikawa).
3. D5/D6 Korrekturmaßnahmen umsetzen und verifizieren.
4. `closure_ready` prüfen → D8 abschließen.

Tipps

- `critical` + offen blockiert oft Release-Readiness.

- `linked_change_request` verbindet Problem mit formalem Change.

---

11. DFMEA / Risiko

Route: `/risks` · **Prozess:** MAN.5 · ISO 26262-3

Zweck

Risikoregister, **DFMEA** (Design-FMEA) und **HARA-lite** (Gefährdungsanalyse) — inkl. **L/M/H-Bewertung** (kein RPN/RPZ, siehe ADR-0026).

Wer darf was?

Operation	Rollen
Lesen	Alle
Schreiben (API)	PE, SAFETY, ADMIN
UI-Schreiben	**Nein** (Lesen)

L/M/H — die maßgebliche DFMEA-Bewertung

> **Wichtig:** HCQ verwendet **keine Risikoprioritätszahl (RPN/RPZ)**. Maßgeblich ist `risk_rating` = **L** (low), **M** (medium), **H** (high).

Serverseitige Ableitung aus S, O, D (jeweils 1–10):

Regel	Bewertung
Schweregrad S ≥ 9	**H** (immer)
Ein Faktor ≥ 7	**H**
Ein Faktor ≥ 5	**M**
Sonst	**L**

S/O/D bleiben Pflichtfelder; `risk_rating` wird **nicht** manuell vom Client gesetzt.

Oberfläche

- **KPI-Karten:** offene Risiken, höchster Score (P×I), DFMEA-Hochbewertungen, Gefährdungen.
- **Tabs:** Risikoregister | DFMEA | HARA

Datenfelder — Risikoregister (`RISK-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`risk_id`	Risiko-ID	`RISK-001`	Ja
`category`	Kategorie	`technical` , `safety` , `ai` , …	Ja
`probability`	Wahrscheinlichkeit (1–5)	`3`	Ja
`impact`	Auswirkung (1–5)	`4`	Ja
`risk_score`	P × I (max. 25)	`12`	Auto
`risk_level`	Stufe	`low` , `medium` , `high`	Auto
`mitigation`	Maßnahme	`Redundanter Sensor`	Ja
`owner`	Verantwortlicher	`safety.lead`	Ja
`status`	Status	`open` … `closed`	Ja

Datenfelder — DFMEA (`DFMEA-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`dfmea_id`	DFMEA-Zeile	`DFMEA-010`	Ja
`function`	Funktion	`Authenticate user`	Ja
`failure_mode`	Fehlermodus	`Token expired early`	Ja
`effect`	Auswirkung	`Unauthorized access`	Ja
`severity` (S)	Schweregrad 1–10	`8`	Ja
`occurrence` (O)	Auftreten 1–10	`3`	Ja
`detection` (D)	Entdeckung 1–10	`2`	Ja
`risk_rating`	**L/M/H**	`high` → **H**	Auto
`recommended_action`	Empfohlene Maßnahme	`Add token refresh`	Ja
`linked_swreq`	Verknüpfte Anforderung	`SWREQ-150`	Nein
`status`	Bearbeitungsstatus	`action_defined`	Ja

Datenfelder — HARA (`HAZ-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`haz_id`	Gefährdungs-ID	`HAZ-001`	Ja
`item`	Item/Funktion	`Braking ECU`	Ja
`hazard`	Gefährdung	`Unintended acceleration`	Ja
`operational_situation`	Betriebssituation	`Highway cruise`	Ja
`severity` (S)	S0–S3	`S3`	Ja
`exposure` (E)	E0–E4	`E4`	Ja
`controllability` (C)	C0–C3	`C2`	Ja
`asil`	ASIL-Klasse	`QM` , `A` , `B` , `C` , `D`	Auto
`safety_goal`	Safety Goal	`SG-001`	Nein
`linked_risk` / `linked_dfmea`	Verknüpfungen	`RISK-001`	Nein

Typischer Workflow

1. DFMEA-Tab → Filter `rating=high` → offene **H**-Einträge priorisieren.
2. `recommended_action` umsetzen → Status `completed` .
3. HARA: ASIL-D-Verteilung im Dashboard beobachten.

Häufige Fehler / Tipps

- **RPN suchen:** Existiert nicht mehr — nur L/M/H verwenden.
- Hohes S mit niedrigem O/D kann trotzdem **H** sein (S≥9-Regel).
- Risikoregister-Score (P×I) ist **unabhängig** von DFMEA L/M/H.

---

12. Change & Configuration

Route: `/changes` · **Prozess:** SUP.8 · SUP.10

Zweck

Änderungsanträge (CR), **Impact-Analyse**, **CCB-Freigaben**, **Konfigurationselemente (CI)** und **Baselines**.

Wer darf was?

Operation	Rollen
Lesen	Alle
Schreiben	CCB, QM, PE, ADMIN
Freigabe (approve)	CCB, QM, ADMIN

Oberfläche

Tabs: Änderungsanträge | Konfigurationselemente | Baselines

- Filter Status/Kategorie bei CRs.
- CR-Zeile → Detail mit Impact, CCB-Entscheidungen, Traceability-Slice.

Datenfelder — Änderungsantrag (`CR-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`cr_id`	Change-ID	`CR-010`	Ja
`title`	Titel	`Update JWT expiry`	Ja
`category`	Kategorie	`code` , `requirement` , `test` , …	Ja
`priority`	Priorität	`high` , `critical`	Ja
`status`	Workflow-Status	`submitted` … `closed`	Ja
`requester`	Antragsteller	`dev.user`	Ja

Datenfelder — Impact-Analyse

Feldname	Bedeutung	Beispiel	Pflicht?
`affected_items`	Betroffene Artefakte	`["SWREQ-150"]`	Ja
`risk_assessment`	Risikobewertung	Freitext	Ja
`effort_estimate`	Aufwandsschätzung	`2 PT`	Ja
`recommendation`	Empfehlung	`approve` , `reject` , `defer`	Ja

Datenfelder — Konfigurationselement (`CI-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`ci_id`	CI-ID	`CI-SRC-API`	Ja
`type`	Typ	`source` , `spec` , `test` , `tool` , `doc`	Ja
`version`	Version	`1.2.0`	Ja
`location`	Speicherort	`src/hcq/api/`	Ja
`baseline_id`	Zugehörige Baseline	`BL-1.0`	Nein
`status`	Status	`draft` , `released` , `obsolete`	Ja

Datenfelder — Baseline (`BL-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`bl_id`	Baseline-ID	`BL-1.0`	Ja
`name`	Name	`Release 1.0 Baseline`	Ja
`item_ids`	Enthaltene CIs	`["CI-SRC-API"]`	Ja
`release_note`	Release-Hinweis	Änderungsübersicht	Nein

Typischer Workflow

1. CR einreichen (`submitted`).
2. Impact-Analyse durch PE/QM.
3. CCB-Freigabe (`approved`) → Implementierung → Verifikation → `closed` .
4. Baseline für Release festlegen.

Tipps

- CR ohne Impact-Analyse: Detail zeigt „Noch keine Impact-Analyse erfasst“.
- `released` CIs sind Freigabe-Grundlage für Releases.

---

13. Releases / Freigabe

Route: `/releases` · **Prozess:** SWE.6 · IATF 8.6

Zweck

Release-Gates mit Kriterien-Checkliste und **RASIC-Sign-offs** vor Freigabe.

Wer darf was?

Operation	Rollen
Lesen	Alle
Sign-off erfassen (UI)	Rollen gemäß RASIC (QM, PE, PM, CCB, CUSTOMER, ADMIN)

Oberfläche

- Filter Status/Typ.
- Release-Zeile → Readiness-Detail:
  - Pflichtkriterien (pass/fail/open).
  - Rollenfreigaben (fehlend/erhalten/abgelehnt) mit **Zeitstempel** (`decided_at` , UTC).
  - Verknüpfte Nachweise (Baseline, Testläufe, offene Probleme, DFMEA-H).
  - **Sign-off erfassen** (nur berechtigte Rolle) — Zeitstempel wird serverseitig gesetzt.

Datenfelder — Release (`REL-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`rel_id`	Release-ID	`REL-1.0`	Ja
`title`	Titel	`HCQ v1.0.0`	Ja
`type`	Typ	`sw_release` , `ppap` , `milestone` , `delivery`	Ja
`target_baseline_id`	Ziel-Baseline	`BL-1.0`	Nein
`status`	Status	`draft` … `released`	Ja

Datenfelder — Kriterium

Feldname	Bedeutung	Beispiel	Pflicht?
`text`	Kriteriumstext	`All tests pass`	Ja
`mandatory`	Pflichtkriterium	`true`	Ja
`result`	Ergebnis	`pass` , `fail` , `open`	Ja

Datenfelder — Sign-off

Feldname	Bedeutung	Beispiel	Pflicht?
`role`	Freigeber-Rolle	`QM`	Ja
`approver_name`	Name	`Anna Schmidt`	Ja
`decision`	Entscheidung	`approve` , `reject`	Ja
`comment`	Kommentar	Optional	Nein
`decided_at`	Zeitstempel (UTC, **serverseitig**, nicht editierbar)	`2026-06-18T14:30:00Z`	Auto

> Die UI zeigt `decided_at` formatiert in der Sign-off-Liste. Der Zeitstempel ist der audit-relevante Nachweis **wann** freigegeben wurde (`SWREQ-016` , RASIC-Freigaben).

Readiness-Felder

Feld	Bedeutung
`can_release`	Alle Gates erfüllt
`criteria_satisfied`	Alle Pflichtkriterien `pass`
`approvals_satisfied`	Alle erforderlichen Rollen haben sign-off
`missing_roles`	Noch ausstehende Rollen
`evidence.dfmea_high_risk_rating`	Anzahl DFMEA mit **H**

Typischer Workflow

1. Release in `in_review` versetzen (API).
2. Kriterien prüfen — rote `fail` beheben.
3. Berechtigte Rollen erfassen Sign-offs.
4. `can_release=true` → Status `released` .

Tipps

- Meldung „Ihre Rolle kann keine Sign-offs erfassen" — RASIC prüfen.
- Offene kritische Probleme blockieren typischerweise `can_release` .

---

14. Rollen & RASIC

Route: `/roles` · **Prozess:** MAN.3 · IATF 5.3

Zweck

RASIC-Matrix (Wer macht was?) und **Rollenregister** für das Projekt.

Wer darf was?

Operation	Rollen
Lesen	Alle
Schreiben (API)	QM, PM, ADMIN
UI-Schreiben	**Nein** (Lesen)

Oberfläche

Tab RASIC-Matrix: Zeilen = Aktivitäten, Spalten = Rollen, Zellen = R/A/S/I/C.

Tab Rollenregister: Code, Name (DE/EN), Beschreibung, System/Benutzerdefiniert.

RASIC-Codes

Code	Bedeutung
R	Responsible — führt aus
A	Accountable — verantwortlich / Freigeber
S	Support — unterstützt
I	Informed — wird informiert
C	Consulted — wird konsultiert

Datenfelder — RASIC-Zuordnung

Feldname	Bedeutung	Beispiel	Pflicht?
`assignment_id`	ID	`RASIC-001`	Ja
`activity`	Aktivität	`release_signoff`	Ja
`role_code`	Rolle	`QM`	Ja
`responsibility`	RASIC-Code	`A`	Ja
`note`	Anmerkung	Kontext	Nein

Datenfelder — Rolle

Feldname	Bedeutung	Beispiel	Pflicht?
`code`	Rollen-Code	`QM`	Ja
`name` / `name_de`	Anzeigename	`Quality Manager`	Ja
`builtin`	Systemrolle	`true`	Ja
`description`	Beschreibung	Verantwortungsbereich	Ja

Typischer Workflow

1. Matrix öffnen → Aktivität „Release Sign-off“ finden.
2. **A**-Rolle identifizieren → diese Person muss Sign-off erfassen.
3. Bei Lücken: QM pflegt Zuordnungen (API).

Tipps

- `approver_roles` pro Aktivitätszeile steuern Release-Freigaben.
- Built-in-Rollen (`ADMIN` , `QM` , …) nicht löschen.

14.1 Benutzerverwaltung & Einladungen (On-Prem)

Route: `/users` · **Prozess:** MAN.3 · IATF 5.3 · **Sichtbar:** nur **ADMIN**
Anforderung: `SWREQ-231` · **API:** `API-ANBINDUNG-KUNDE.md` §5.1

Zweck

On-Premise-Installationen haben **keine offene Self-Registration**. Der Administrator legt Benutzer per **Einladungslink** an — ohne das Passwort zu kennen. Air-gap-tauglich: Link per Teams, USB oder internem Ticket weitergeben (HCQ versendet **keine** E-Mails).

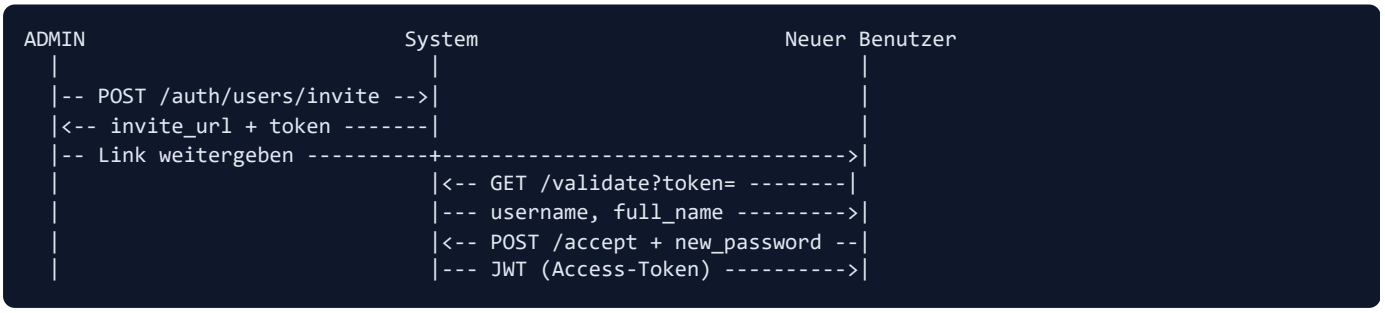
Wer darf was?

Operation	Rollen	UI / API
Benutzerliste	ADMIN	`GET /auth/users`, UI Tab „Registrierte Benutzer“
Benutzer einladen	ADMIN	`POST /auth/users/invite`, UI Formular
Benutzer anlegen (mit Passwort)	ADMIN	`POST /auth/users` (Legacy/API — Admin kennt Passwort)
Einladung annehmen	Öffentlich (Token)	`GET /auth/invite/validate`, `POST /auth/invite/accept`, UI `/invite?token=…`

Oberfläche (Admin)

Element	Bedeutung
Benutzer einladen	Formular: Benutzername, Anzeigename, Rolle
Einladung erstellen	Legt Account an + zeigt **Einmal-Link** und Ablaufzeit
Link kopieren	Link wird nur **einmal** angezeigt — sicher archivieren/weitergeben
Benutzerliste	Status: Aktiv / Einladung offen (`must_change_password`)

Ablauf Einladung (End-to-End)



API-Endpunkte

Methode	Pfad	Auth	Beschreibung
`POST`	`/auth/users/invite`	ADMIN	Benutzer + Token/Link anlegen
`GET`	`/auth/invite/validate?token=`	öffentlich	Einladung prüfen
`POST`	`/auth/invite/accept`	öffentlich	Passwort setzen + JWT

Request Einladung:

```
{
  "username": "pe_mueller",
  "full_name": "PE Müller",
  "role": "PE"
}
```

Response (Token nur einmal):

```
{
  "invite_token": "...",
  "invite_url": "https://hcq.firma.local/invite?token=...",
  "expires_at": "2026-07-03T12:00:00+00:00",
  "user": { "username": "pe_mueller", "must_change_password": true, ... }
}
```

Konfiguration (Umgebungsvariablen)

Variable	Default	Bedeutung
`HCQ_PUBLIC_APP_URL`	`http://localhost:5173`	Basis-URL für generierte Einladungslinks
`HCQ_INVITE_TOKEN_EXPIRE_HOURS`	`72`	Gültigkeit des Einmal-Tokens (Stunden)

Sicherheit & Audit

- Invite-Token wird **nur gehasht** (SHA-256) in der DB gespeichert.
- **Single-Use:** Nach Annahme ungültig; abgelaufene Tokens → `401`.
- Audit-Trail: `user_invited`, `invite_accepted` unter `GET /audit`.
- RBAC: Pfad `/auth/users/*` → Ressource `users`, nur Rolle **ADMIN**.

Typischer Workflow (On-Prem)

1. Als **ADMIN** anmelden → **Governance** → **Benutzer**.
2. Benutzername (z. B. `qm_fischer`), Name, Rolle **QM** → **Einladung erstellen**.
3. Link kopieren und an Kollegin senden (intern).
4. Kollegin öffnet Link, setzt Passwort, arbeitet in HCQ.
5. Admin prüft in der Liste: Status wechselt von „Einladung offen“ zu „Aktiv“.

Häufige Fehler

Problem	Ursache	Lösung
Link „ungültig/abgelaufen“	>72 h oder bereits genutzt	Neue Einladung erstellen
403 beim Einladen	Kein ADMIN	Mit Admin-Account anmelden
409 Benutzer existiert	Username vergeben	Anderen Namen oder alten Account prüfen
Falscher Link-Host	`HCQ_PUBLIC_APP_URL` falsch	Env vor Start setzen

Qualifikation / Tests

Testfall	Datei
Admin-Einladung, Validate, Accept, JWT	`tests/test_auth_invite.py`
RBAC, Single-Use, Ablauf, Duplikat	`tests/test_auth_invite.py`
Protokoll	`docs/quality/test-results/QT-229-231-backtest.md`

---

15. Security Events

Route: `/security-events` · **Prozess:** SUP.10 · SIEM · **Sichtbar:** ADMIN, QM

Zweck

SIEM-tauglicher Sicherheitsereignis-Feed (Logins, MFA, AuthZ-Verweigerungen, Admin-Aktionen).

Wer darf was?

Operation	Rollen
Lesen & Export	ADMIN, QM

Oberfläche

- Filter **Ereignistyp**, Zeitraum.
- Tabelle: Zeitstempel, Typ, Schweregrad, Akteur, Quell-IP, Details.
- **Export:** JSON, JSONL, CEF.
- Paginierung.

Datenfelder

Feldname	Bedeutung	Beispiel	Pflicht?
`timestamp`	Zeitpunkt (UTC)	ISO-8601	Ja
`event_type`	Typ	`login_success` , `authz_denied` , `mfa_challenge` , …	Ja
`severity`	Schweregrad	`info` … `critical`	Ja
`actor`	Benutzer	`admin`	Ja
`source_ip`	IP-Adresse	`192.168.1.10`	Nein
`source`	Datenquelle	`audit_trail` , `security_events`	Ja
`details`	Zusatzdaten (JSON)	Kontext	Nein

Typischer Workflow

1. Nach Sicherheitsvorfall: Filter `login_failure` / `authz_denied` .
2. Akteur und IP korrelieren.
3. JSONL/CEF an SIEM exportieren.

Tipps

- Modul in Navigation nur sichtbar für ADMIN/QM — kein Fehler, sondern RBAC.
- Export großer Zeiträume kann dauern — Filter eng setzen.

---

16. Compliance / Norm-Matrix

Route: `/compliance` · **Prozess:** Multi-Norm (IATF, ISO 26262, VDA, ASPICE, Robotik-Roadmap)

Branchenfilter (`SWREQ-229`)

Die angezeigten Standards hängen von der **Branchenwahl** ab (Login/Kopfzeile, siehe §2.1a):

Domain	Standards (Tabs / KPI)	Reifegrad
automotive	ASPICE v4.0, IATF 16949, ISO 26262, VDA 6.3	Kuratiert, ehrlicher Status
robotics	ISO 10218, ISO/TS 15066, ISO 13849, IEC 62443	**Roadmap-Scaffold** — Disclaimer in UI

API: `GET /compliance/overview?domain=automotive|robotics` , Export analog mit `?domain=` .

Zweck

Mehrnormiger Audit-Einstieg: Zuordnung von Norm-Klauseln zu HCQ-Nachweisen mit ehrlichem Status (kein Zertifikatsanspruch).

Wer darf was?

Lesen: alle. Schreiben: **Nein** (kuratiertes Mapping, read-only).

Oberfläche

- KPI-Karten pro Standard + Gesamtabdeckung.
- **Audit-Posture**-Text mit Disclaimer.
- **Tabs:** ASPICE v4.0 | IATF 16949 | ISO 26262 | VDA 6.3
- Filter Status pro Standard.
- **JSON exportieren** — vollständige Matrix.

Datenfelder — Klausel

Feldname	Bedeutung	Beispiel	Pflicht?
`standard`	Norm	`aspice_v40`	Ja
`clause_id`	Klausel-ID	`SWE.1`	Ja
`title`	Titel	Software Requirements Analysis	Ja
`status`	Erfüllungsgrad	`fulfilled` , `partial` , `roadmap` , `not_applicable`	Ja
`hcq_evidence`	Nachweis-Referenzen	`["SWREQ-010", "QT-010"]`	Ja
`note`	Erläuterung	Lücken, Roadmap	Ja
`basis`	Datenbasis	`curated` , `live`	Ja

Status-Bedeutung

Status	Bedeutung
`fulfilled`	Nachweis im Repo vorhanden
`partial`	Teilweise umgesetzt
`roadmap`	Geplant, noch nicht erfüllt
`not_applicable`	Nicht anwendbar (tailoring)

Typischer Workflow

1. Standard-Tab wählen (z. B. ASPICE).
2. Filter `partial` / `roadmap` → Gap-Liste für Roadmap.
3. `hcq_evidence` anklicken mental → RTM/Tests prüfen.
4. JSON für Audit-Paket exportieren.

Tipps

- Disclaimer lesen: **Selbstbewertung**, keine Zertifizierung.
- `live` -Zeilen werden aus echten Zählern angereichert; `curated` ist redaktionell.
- Bei **Robotik**: Roadmap-Banner lesen — kein vollständiges Mapping wie Automotive.

---

17. On-Prem-Lizenz & Add-ons

Bezug: `ON-PREM-LIZENZ.md` · ADR-0039 · SWREQ-251..252
UI: Framework-Gate beim Login · Add-on-Freischaltung unter `/profile` oder auf Add-on-Seiten

Zweck

On-Premise-Kunden erhalten HCQ **offline** per signiertem JWT (RS256). Die Lizenz steuert, welche Module sichtbar und nutzbar sind — ohne Cloud-Telemetrie. HCQ **stellt keine Zertifizierung** aus; die Lizenz ist ein **technischer Freischaltmechanismus**, kein Compliance-Nachweis.

JWT-Features (Token-Inhalt)

Feature im JWT	Wirkung
`framework`	Gesamte HCQ-UI + API (bei aktivem Enforcement)
`digital_twin`	Navigation **Digital Twin**, Validierung, Monatsabschluss
`cra`	**CRA-Nachweisführung** (`/cra`), Readiness aus Digital-Twin-Profilen

Digital Twin und CRA setzen eine gültige **Framework**-Lizenz voraus, wenn `LICENSE_ENFORCEMENT_ENABLED=true` . CRA setzt zusätzlich Digital-Twin-Daten im Silo voraus — ohne Twins liefert `/cra/status` leere Metriken, aber keinen Fehler.

Laufzeiten

Term	Tage	Typischer Einsatz
`3m`	90	Pilot / Audit-Sprint
`12m`	365	Jahreslizenz inkl. Add-ons

UI-Pfade — Freischaltung

Schritt	Wo	Aktion
1	Login-Screen	Bei Enforcement: **Framework-Token** eingeben (`FrameworkLicenseGate`)
2	`/profile` → **Add-on-Freischaltung**	Token mit `digital_twin` bzw. `cra` einlösen
3	`/digital-twin` oder `/cra`	Ohne passendes Feature: `AddonUnlockCard` mit Token-Eingabe
4	Sidebar	Nav-Punkte **Digital Twin** / **CRA** erscheinen erst nach Freischaltung

Das JWT wird **nur im Browser** (`localStorage`) gespeichert und als Header `X-License-Token` an die API gesendet. Abgelaufene Tokens: HTTP 403, Code `LICENSE_REQUIRED` .

Simulation / Demo (Auto-Unlock)

Konfiguration	Framework	Digital Twin	CRA
Standard `ENVIRONMENT=simulation` , `LICENSE_ENFORCEMENT_ENABLED=false`	offen	frei wenn `DIGITAL_TWIN_MODULE_ENABLED=true`	frei wenn `CRA_MODULE_ENABLED=true`
Demo mit `LICENSE_ENFORCEMENT_ENABLED=true`	Token nötig	Token mit Feature	Token mit `cra`

Demo-Tokens: `data/demo/license-demo-12m.token` (Framework + DT), `data/demo/license-demo-12m-full.token` (+ CRA). Details: `data/demo/LICENSE-TOKENS.md` .

Umgebungsvariablen (Kurz)

Variable	Bedeutung
`LICENSE_ENFORCEMENT_ENABLED`	`true` = API blockiert ohne gültiges `framework` -JWT
`LICENSE_PUBLIC_KEY_PEM`	RS256 Public Key (Vendor liefert; Simulation: DEV-Key)
`DIGITAL_TWIN_MODULE_ENABLED`	Modul überhaupt anbieten
`CRA_MODULE_ENABLED`	CRA-Add-on überhaupt anbieten

API (Kurzreferenz)

Methode	Pfad	Zweck
GET	`/license/status`	Enforcement, Restlaufzeit, Feature-Flags
POST	`/license/unlock`	Token validieren (UI)
GET	`/digital-twin/status`	Erfordert `digital_twin` im Token
GET	`/cra/status`	Erfordert `cra` im Token

Vendor-CLI: `hcq-license keygen` , `issue` , `verify` — siehe `ON-PREM-LIZENZ.md` .

Tipps

- **Privaten Signing-Key** nie ins Kunden-Image oder Repo legen.
- Add-on-Token kann Framework + mehrere Features in einem JWT bündeln.
- In der öffentlichen Demo ohne Enforcement bleibt der Framework-Gate ausgeblendet.

---

18. Digital Twin

Route: `/digital-twin` · **CLI:** `hcq-dt` · **Prozess:** SWE.2, SUP.8, EU CRA (Werkzeug)
Bezug: `digital-twin-on-prem.md` · ADR-0039 · SWREQ-245..250, SWREQ-254

Zweck

Der **Digital Twin** dokumentiert Typ- und Instanz-Twins im **Kundensilo** (On-Prem / Air-Gap). HCQ liefert Schema, Validator und Monatsabschluss — der Kunde speichert und attestiert die Daten **selbst**. Keine permanente HCQ-Cloud-Speicherung.

> **Disclaimer:** Self-Attestation im Kundensilo — **kein Konformitätszertifikat** und keine CRA-Konformitätsbehauptung durch HCQ (SWREQ-250).

Wer darf was?

Operation	Rollen	UI / API
Status lesen	Alle (mit DT-Lizenz)	`GET /digital-twin/status` , UI KPI-Karten
Monatsabschluss	QM, ADMIN (API); UI-Button	`POST /digital-twin/month-close` , `hcq-dt month-close`
Attestations-Ledger lesen	Alle (mit DT-Lizenz)	`GET /digital-twin/attestations`
Twin erzeugen/validieren	IT/QM (CLI)	`hcq-dt validate` , `generate-type` , `generate-instance`

Navigation & Sichtbarkeit

1. Sidebar **Qualität** → **Digital Twin** (nur wenn `DIGITAL_TWIN_MODULE_ENABLED` und Token-Feature `digital_twin` freigeschaltet).
2. Ohne Modul-Flag: Hinweiskarte „nicht lizenziert".
3. Ohne Token: `AddonUnlockCard` auf der Seite oder unter `/profile`.

Oberfläche (`/digital-twin`)

Bereich	Inhalt
Validierungsstatus	Ampel gültig/ungültig, Issue-Zähler
Zähler	Typ-Twins, Instanz-Twins (Stromzähler-Modell)
Registry-Hash	Integritätsanker des Twin-Registers
Befunde	Tabelle offener Validierungsregeln
Monatsabschluss	Periode `YYYY-MM` wählen → Self-Attestation erzeugen
Attestations-Ledger	Timeline vergangener Monatsabschlüsse

Attestations-Ledger (Timeline, SWREQ-254)

Nach jedem Monatsabschluss kann ein Eintrag **append-only** im Silo-SQLite persistiert werden.

Feld	Bedeutung
`period`	Abrechnungsperiode `YYYY-MM` (eindeutig)
`type_count` / `instance_count`	Zählerstände zum Abschluss
`registry_hash`	Integritätsanker nach Abschluss
`previous_hash`	Hash der Vorgängerperiode (Kettenglied)
`registry_hash_changed`	Abweichung zum Vorgänger (Readiness-Hinweis)
`created_at`	Zeitstempel der Persistierung

UI: Tabelle unterhalb des Monatsabschluss-Formulars mit Spalten Periode, Zähler, Hash, Hash-Änderung. Disclaimer unter der Timeline: formaler Audit-Trail, **kein Zertifikat**.

CLI: `hcq-dt attestations list [--json] [--db-path <sqlite>]`

CLI `hcq-dt` (Operator)

Befehl	Zweck
`hcq-dt validate`	Registry gegen Schema prüfen
`hcq-dt generate-type --bauteil BT-005`	Typ-Twin erzeugen
`hcq-dt generate-instance --serial …`	Instanz-Twin erzeugen
`hcq-dt month-close --period 2026-06`	Monatsattestierung (JSON)
`hcq-dt attestations list`	Ledger-Historie
`hcq-dt attest-import <file>`	Kunden-Submission prüfen (Stichprobe)

Vollständiger Prozess: `digital-twin-on-prem.md`.

Typischer Workflow (QM/IT)

1. BaC/BIT validieren (`hcq-bac validate` , `hcq-bit validate`).
2. Twins erzeugen und im Kunden-QMS archivieren.
3. UI `/digital-twin` → Validierung und Zähler prüfen.
4. Monatsende: Periode wählen → **Monatsabschluss** → Ledger-Eintrag prüfen.
5. Optional: `registry_hash` intern an Audit weitergeben (ohne Seriennummern-Export).

Roadmap (geplant, nicht verfügbar)

Funktion	Status
SBOM-Vollmodul / NTIA-Validator-UI	**geplant**
MES-Massen-Sync	**geplant**
Secure Import Gateway (signierte Update-Pakete)	**geplant** (ADR-0041)

---

19. CRA-Nachweisführung

Route: `/cra` · **API:** `/cra/*` · **Prozess:** EU CRA Readiness (Werkzeug)
Bezug: `cra-evidence-chain.md` · SWREQ-252, SWREQ-264

Zweck

Das **CRA-Add-on** zeigt **Readiness** und **Lücken** aus Digital-Twin-Profilen (DT-003, DT-005) — strukturierte Nachweisführung im Kundensilo.

> **Disclaimer (verbindlich):** CRA-Ansichten sind **Lückenidentifikation und Readiness** — **keine CRA-Konformität**, kein Konformitätszertifikat, keine Zertifizierung durch HCQ. Die Einhaltung der EU-Verordnung (EU) 2024/1687 obliegt allein dem Hersteller.

Wer darf was?

Operation	Rollen	UI / API
Status lesen	Alle (mit CRA-Lizenz)	`GET /cra/status` , UI
Gaps pro Bauteil	Alle (mit CRA-Lizenz)	`GET /cra/gaps-by-bauteil`
Baseline erfassen	QM, ADMIN	`POST /cra/baseline` (vor Ruleset-Update)

CRA erfordert JWT-Feature `cra` und `CRA_MODULE_ENABLED=true` . Ohne Digital-Twin-Daten: leere Metriken, kein Fehler.

Navigation

Sidebar **Qualität** → **CRA** (Add-on, gleiche Freischaltlogik wie Digital Twin). Freischaltung: `/profile` oder `AddonUnlockCard` auf `/cra` .

Oberfläche (`/cra`)

Bereich	Inhalt
Readiness-Ampel	`valid` = keine CRA-relevanten Lücken; sonst „Lücken offen"
KPI-Karten	Digitale Produkttypen, `shipped_without_sbom` , `ruleset_version`
Warnung neu nicht konform	Banner wenn `newly_non_compliant_count > 0`
Lücken pro Bauteil	Tabelle `BT-*` mit Compliance, Regeln, `newly_non_compliant` -Badge
Gesamt-Lückenliste	Alle Issues mit Severity, Regel, Meldung
Disclaimer	SWREQ-250-Text unter den KPIs

Baseline vor Ruleset-Update (SWREQ-264)

Vor dem Import eines neuen Validator-Rulesets soll der Operator eine **Baseline** erfassen:

```
POST /cra/baseline
```

Speichert `ruleset_version` , `registry_hash` und Compliance-Snapshot pro Bauteil im Silo. Nach dem Update vergleicht die API Bauteile mit `newly_non_compliant: true` — Bauteile, die zuvor konform waren und durch strengere Regeln nun Lücken zeigen.

UI: Warnbanner mit Anzahl und `baseline_captured_at` , Badge „neu nicht konform" in der Bauteil-Tabelle. **Baseline per API** — dedizierte UI-Schaltfläche folgt als Increment.

API (Kurzreferenz)

Methode	Pfad	Zweck
GET	`/cra/status`	Readiness-Snapshot, Metriken, Issues, `newly_non_compliant_count`
GET	`/cra/gaps-by-bauteil`	Aggregation pro `BT-*`
POST	`/cra/baseline`	Baseline vor Ruleset-Update

Report wird **on-demand** berechnet (kein Hintergrund-Cache) — Air-Gap-tauglich.

Typischer Workflow

1. Digital Twins pflegen und validieren (§18).
2. Vor HCQ-Update mit neuem Ruleset: **`POST /cra/baseline`** .
3. Update importieren (Roadmap: Secure Import Gateway).
4. `/cra` öffnen → Refresh → `newly_non_compliant` prüfen.
5. Lücken im QMS dokumentieren; Monatsabschluss verankert post-Import-Hash (§18).

Roadmap (geplant)

Funktion	Status
CRA-Nachweis-Export (ZIP/JSON-Bundle)	**geplant** (SWREQ-261)
Attention-Ledger-Timeline in UI	**geplant** (SWREQ-260)
VDP (CVE-Katalog) UI	**geplant**

20. Zertifikatsregister

Route: `/certificates` · **Prozess:** SUP.1, IATF 7.2 (Kompetenznachweise)

Bezug: ADR-0040 · SWREQ-253

Zweck

Metadaten-Register für **Kunden-Zertifikate und Nachweise** (TLS, Produkt, Audit, Sonstiges) mit **90-Tage-Ablauf-Warnung**. HCQ ist **kein PKI/HSM** — es verwaltet nur Registerdaten, keine privaten Schlüssel oder Zertifikatsketten.

> **Disclaimer:** Kein Ersatz für externes Zertifizierungsaudit. HCQ stellt **keine** Zertifizierung dar.

Wer darf was?

Operation	Rollen
Lesen	Alle authentifizierten Rollen
Anlegen / Ändern / Archivieren	QM, PM, ADMIN

Oberfläche (`/certificates`)

Bereich	Inhalt
Disclaimer	Abgrenzung Metadaten-Register (oben auf der Seite)
KPI-Karten	Gesamt, aktiv, **bald ablaufend**, abgelaufen, archiviert
Filter	Status, Typ, Ablauf-Hinweis (`ok` / `warning` / `expired`)
Tabelle	Zertifikats-ID, Typ, Gültigkeit, Ablauf-Badge, Verantwortlicher

90-Tage-Warnung

`expiry_alert`	Bedingung	UI
`ok`	Mehr als 90 Tage bis `valid_until`	Grünes Badge
`warning`	≤ 90 Tage, noch nicht abgelaufen	Gelbes/Warn-Badge, KPI „bald ablaufend"
`expired`	`valid_until` überschritten	Rotes Badge

Die Schwelle ist fest (**90 Kalendertage**, `_WARNING_DAYS` im Backend) — nicht konfigurierbar in der UI.

Datenfelder (Auszug)

Feld	Bedeutung	Pflicht?
`certificate_id`	Eindeutige ID	Ja
`type`	`tls` , `product` , `audit` , `other`	Ja
`name`	Bezeichnung	Ja
`valid_from` / `valid_until`	Gültigkeitszeitraum	Ja
`status`	`active` , `expired` , `revoked` , `archived`	Ja
`expiry_alert`	Abgeleitet (`ok` / `warning` / `expired`)	Auto
`days_until_expiry`	Resttage	Auto

Typischer Workflow (QM)

1. Register öffnen → KPI **bald ablaufend** prüfen.
2. Filter `warning` → betroffene Zertifikate erneuern lassen.
3. Nach Erneuerung: Eintrag aktualisieren oder neuen anlegen, alten archivieren.

---

21. CRA-Nachweiskette & Compliance-Stack

Bezug: `cra-evidence-chain.md` · `HCQ-PRODUKT-COMPLIANCE-PAKET.md` · ADR-0041, ADR-0044 · SWREQ-261, SWREQ-255

Zweck

Signierte Updates (Rulesets, CVE-Feeds, SBOM-Pakete) sollen in On-Prem-Silos in eine **auditierbare Kette** eingebunden werden: Import → Twin-Zustand → CRA-Readiness → Monatsattestierung. Dieses Kapitel verknüpft die UI-Module mit dem Prozessleitfaden.

> **Disclaimer:** Die gesamte Kette dokumentiert **Nachweise und Lücken** im Kundensilo — **Readiness, nicht Konformität**. HCQ belegt nicht die Einhaltung der EU CRA-Verordnung.

Kette (Operator-Lesart)

```
Trusted Update (USB/CDN) → Import-GW (Signaturprüfung) → DT-Validator / CRA-Report
    → Monatsabschluss → Attestations-Ledger (registry_hash verankert)
```

Stufe	Modul / Artefakt	Handbuch
Lizenz & Silo	On-Prem JWT	§17
Twin-Daten	Digital Twin	§18
CRA-Readiness	`/cra/status`, gaps-by-bauteil	§19
Zertifikate (parallel)	`/certificates`	§20
Norm-Matrix (parallel)	`/compliance`	§16
Produkt-Dogfooding	Compliance-Paket	`HCQ-PRODUKT-COMPLIANCE-PAKET.md`

Operator-Checkliste nach Import

1. Import-Receipt prüfen (Signatur OK) — **geplant:** Secure Import Gateway.
2. `POST /cra/baseline` — Baseline vor Ruleset-Update (§19).
3. `hcq-dt validate` — offene Findings dokumentieren.
4. `/cra` → Refresh → `newly_non_compliant` beachten.
5. Periodenende: Monatsabschluss → Ledger prüfen (§18).
6. Optional: Diff `registry_hash_changed` im QMS vermerken.

Roadmap-Komponenten (geplant, nicht als verfügbar darstellen)

Komponente	ADR	Status
Secure Import Gateway	ADR-0041	**geplant**
SBOM Generator / Store (ARC-SBOM)	—	**geplant**
VDP / CVE-Katalog (ARC-VDP)	—	**geplant**
Attention Ledger (UI-Timeline)	ADR-0044	**geplant**
CRA-Nachweis-Export-Bundle	SWREQ-261	**geplant**

Detail: `cra-evidence-chain.md` · `digital-twin-on-prem.md` §7.

Verwandte Dokumente

Dokument	Inhalt
`ON-PREM-LIZENZ.md`	JWT, Vendor-CLI, Docker-Variablen
`digital-twin-on-prem.md`	Monatsattestierung, RACI
`cra-evidence-chain.md`	End-to-End-Kette, Artefakte
`HCQ-PRODUKT-COMPLIANCE-PAKET.md`	HCQ als Produkt — Dogfooding, Operator-Checkliste §8
`HCQ-EVIDENZ-MATRIX.md`	Requirement → Artefakt → Test → Status
`HCQ-CRA-SELBSTBEWERTUNG.md`	CRA-Vendor-Rolle (Readiness, kein Zertifikat)
`COMPLIANCE-STATUS.md`	Master-Reifegrad je Norm

---

22. KI-Governance

Route: `/ai-governance` · **Prozess:** ISO 42001 · EU AI Act

Zweck

Operative Nachweise für **KI-Systeme** in HCQ: Registry, Model Cards, Datasets, Monitoring, ISO-42001-Checkliste.

Wer darf was?

Operation	Rollen
Lesen	Alle
KI-System anlegen (UI)	QM, ADMIN
Weitere CRUD (API)	QM, ADMIN
Compliance-Checkliste	**Read-only** (alle)

Oberfläche ehrlich

Tab	UI-Funktion
Registry	Liste + Filter; **„KI-System registrieren“** (QM/ADMIN)
Model Cards	**Nur Lesen** (API kann anlegen)
Datasets	**Nur Lesen**
Monitoring	**Nur Lesen**
Compliance	Read-only Checkliste mit Status

Datenfelder — KI-System (`AIS-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`system_id`	System-ID	`AIS-001`	Ja (auto)
`name`	Name	`HCQ Assistant`	Ja
`purpose`	Zweck	`Qualitätsberatung`	Ja
`eu_risk_class`	EU-AI-Act-Risiko	`minimal` , `high` , …	Ja
`owner`	Verantwortlicher	`QM`	Ja
`status`	Status	`draft` , `active` , `retired`	Ja
`version`	Version	`1.0`	Ja

Datenfelder — KI-Risiko (AIIA-lite, `AIR-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`risk_type`	Typ	`bias` , `drift` , `privacy` , …	Ja
`probability` / `impact`	1–5	`3` , `4`	Ja
`risk_score`	P × I	`12`	Auto
`treatment`	Behandlung	Maßnahmenplan	Ja
`linked_risk_id`	Link zum Risikoregister	`RISK-005`	Nein

Datenfelder — Model Card (`MC-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`model_name`	Modellname	`hcq-embed-v1`	Ja
`intended_use`	Vorgesehene Nutzung	Embeddings für Suche	Ja
`out_of_scope_use`	Ausgeschlossene Nutzung	Medizinische Diagnose	Ja
`limitations`	Grenzen	Kein Echtzeit-Betrieb	Ja
`human_oversight`	Menschliche Aufsicht	QM reviewt Outputs	Ja
`performance_metrics`	Kennzahlen	`{"accuracy": "0.92"}`	Nein

Datenfelder — Dataset (`DS-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`name`	Name	`requirements-corpus`	Ja
`provenance`	Herkunft	Interne Docs	Ja
`pii_flag`	Personenbezogene Daten	`true` / `false`	Ja
`bias_analysis`	Bias-Analyse	Freitext	Ja
`license`	Lizenz	`Proprietary`	Ja

Datenfelder — Monitoring-Event (`AIME-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`event_type`	Typ	`drift` , `incident` , `bias_alert`	Ja
`severity`	Schweregrad	`low` … `critical`	Ja
`summary`	Kurzbeschreibung	`Accuracy drop 5%`	Ja

Typischer Workflow

1. Neues KI-System in Registry anlegen (QM).
2. Model Card + Dataset per API dokumentieren.
3. Monitoring-Events bei Vorfällen erfassen (API).
4. Compliance-Tab → offene Controls in Roadmap aufnehmen.

Tipps

- Posture-Disclaimer: **kein ISO-42001-Zertifikat**.
- Monitoring ist **manuell** — kein Auto-Drift-Detection in MVP.

---

23. APQP / PPAP

Route: `/apqp` · **Prozess:** SYS.1 · SYS.5

Zweck

APQP-Phasen mit Lieferobjekten und Gate-Kriterien sowie **PPAP-Pakete** (18 Elemente) und **PSW** (Part Submission Warrant).

Wer darf was?

Operation	Rollen
Lesen	Alle
Schreiben (API)	PE, PM, QM, ADMIN

Oberfläche

- KPI: Phasen, Lieferobjekte, PPAP-Pakete.
- **Tab APQP-Phasen** — Phasen mit Status, Deliverables, Gate-Kriterien.
- **Tab PPAP-Pakete** — Filter Status; Detail mit 18 Elementen + PSW.

Datenfelder — APQP-Phase

Feldname	Bedeutung	Beispiel	Pflicht?
`phase_id`	Phasen-ID	`APQP-P1`	Ja
`name`	Name	`Plan and Define`	Ja
`gate_criteria`	Gate-Kriterien	Freitext	Ja
`status`	Status	`in_progress` , `completed`	Ja
`deliverables[].name`	Lieferobjekt	`DFMEA`	Ja
`deliverables[].status`	Status	`complete`	Ja

Datenfelder — PPAP-Paket (`PPAP-*`)

Feldname	Bedeutung	Beispiel	Pflicht?
`ppap_id`	PPAP-ID	`PPAP-001`	Ja
`part_number`	Sachnummer	`12345-ABC`	Ja
`part_name`	Teilename	`Bracket`	Ja
`customer`	Kunde	`OEM XY`	Ja
`level`	Einreichungsstufe (1–5)	`3`	Ja
`status`	Paketstatus	`submitted` , `approved`	Ja
`elements[].number`	Element-Nr. (1–18)	`1`	Ja
`elements[].status`	Elementstatus	`approved`	Ja
`elements[].evidence_ref`	Nachweis	`DOC-001`	Nein

Datenfelder — PSW

Feldname	Bedeutung	Beispiel	Pflicht?
`authorized_by`	Autorisiert durch	`Quality Director`	Ja
`results_meet_requirements`	Ergebnisse OK	`true`	Ja
`submission_level`	PPAP-Level	`3`	Ja
`status`	PSW-Status	`approved`	Ja

Typischer Workflow

1. APQP-Phasen durcharbeiten → Deliverables auf `complete` .
2. PPAP-Paket anlegen → 18 Elemente befüllen.
3. PSW ausfüllen → `approved` → an Kunden übergeben.

Tipps

- `not_applicable` für Elemente nutzen, die beim Level nicht gefordert sind.
- Phasenabschluss-Ratio im Dashboard beobachten.

---

24. IATF 16949

Route: `/iatf` · **Prozess:** SUP.1 · MAN.5 · MAN.6 · **Traceability:** SWREQ-100…102 · **Doku-Landkarte:** `docs/quality/iatf16949-documentation-map.md`

Zweck

IATF-16949-Compliance-Checkliste, **QMS-KPIs** und **NCR** (Non-Conformance Reports) mit **Klausel-Heatmap** (Rollup Abschnitte 4–10) und **Audit-Readiness-Score**. HCQ bildet einen **repräsentativen Klauselausschnitt** ab — kein Ersatz für ein formales IATF-Zertifikat (siehe Landkarte §1).

Wer darf was?

Operation	Rollen
Lesen (UI + API)	Alle authentifizierten Rollen
Schreiben (API)	QM, PM, ADMIN

Oberfläche

- **Kopfzeile (StatCards):** Konformität %, Audit-Bereitschaft, offene NCRs, KPIs außerhalb Ziel.
- **Tab Checkliste:** Klausel-Heatmap (4–10) + filterbare Tabelle aller Positionen.
- **Tab KPIs:** Kartenraster mit Ziel/Ist, Trend und `on_target`.
- **Tab NCRs:** Tabelle mit Schweregrad, Status und Verknüpfungen.

Die Heatmap färbt jede oberste Klausel nach dem Anteil konformer (nicht-n.a.) Positionen: grün ≥ 80 %, gelb ≥ 50 %, rot darunter, grau wenn keine Positionen.

Datenfelder — Checklistenpunkt (`IATF-CL-###`)

Feldname	Bedeutung	Beispiel	Pflicht?
`item_id`	Punkt-ID (automatisch)	`IATF-CL-001`	Ja (read-only)
`clause`	IATF-Klauselpfad	`8.3.3` , `9.1.1`	Ja
`title`	Kurztitel der Anforderung	Design and development inputs	Ja
`status`	Konformität	`compliant` , `partial` , `non_compliant` , `not_applicable`	Ja
`evidence_ref`	Verweis auf Nachweis (Doku, Ticket, Audit-Report)	`DOC-QMS-01` , `ADR-0013`	Nein

Status-Semantik: `not_applicable` zählt weder in Zähler noch Nenner der Compliance-%. `partial` und `non_compliant` senken den Grad und die Audit-Readiness.

Datenfelder — KPI (`IATF-KPI-###`)

Feldname	Bedeutung	Beispiel	Pflicht?
`kpi_id`	KPI-ID (automatisch)	`IATF-KPI-001`	Ja (read-only)
`metric`	Kennzahl	`Customer complaints` , `On-time delivery`	Ja
`target` / `actual`	Ziel / Ist	`10` / `8` oder `95.0` / `97.0`	Ja
`unit`	Einheit	`count` , `%`	Ja
`period`	Berichtsperiode	`2026-Q2` , `2026-06`	Ja
`higher_is_better`	Zielrichtung	`false` = weniger ist besser; `true` = mehr ist besser	Ja
`trend`	Trend vs. Vorperiode	`improving` , `stable` , `declining`	Ja (bei Anlage)
`on_target`	Im Ziel?	`true` / `false`	Auto (serverseitig)

`on_target` wird beim Anlegen und bei jeder Änderung von Ziel/Ist neu berechnet — nicht manuell setzbar.

Datenfelder — NCR (`IATF-NCR-###`)

Feldname	Bedeutung	Beispiel	Pflicht?
`ncr_id`	NCR-ID (automatisch)	`IATF-NCR-001`	Ja (read-only)
`title`	Kurztitel	Calibration overdue	Ja
`description`	Sachverhalt	Gauge calibration interval exceeded on line 2.	Ja
`clause`	Bezugsklausel	`10.2` , `7.1.5`	Nein
`severity`	Schweregrad	`minor` , `major`	Ja
`status`	Bearbeitungsstatus	`open` , `contained` , `corrected` , `closed`	Ja
`linked_problem`	Verknüpfter 8D-Problembericht	`PR-001`	Nein
`linked_item`	Verknüpfte Checklisten-Position	`IATF-CL-003`	Nein

`linked_problem` wird nur formatvalidiert (`PR-###`); `linked_item` muss existieren.

Audit-Readiness (`GET /iatf/audit-readiness`)

Feld	Bedeutung
`readiness_score`	0–100 (heuristisch aus Compliance, NCR, KPI)
`audit_ready`	`true` nur wenn Schwellen erfüllt **und** keine offenen Major-NCRs
`compliance_ratio`	Anteil konformer anwendbarer Checklistenpunkte
`major_open_ncrs`	Blocker: bei > 0 ist `audit_ready` typischerweise `false`
`open_ncrs`	Gesamtzahl offener NCRs
`non_compliant_items`	Anzahl `non_compliant` -Positionen
`kpis_off_target`	Anzahl KPIs mit `on_target=false`

Typische Workflows

A — Audit-Checkliste aufbauen

1. Pro relevante IATF-Unterklausel eine Position anlegen (`POST /iatf/checklist`).
2. `evidence_ref` auf Repo-Nachweis setzen (z. B. Prozess-Doku, Test-Report).
3. Heatmap prüfen: welche obersten Klauseln (4–10) sind rot/gelb?
4. `GET /iatf/summary` für den Gesamt-Compliance-Grad.

B — KPI-Monitoring (MAN.6 / IATF 9.1)

1. KPIs pro Periode anlegen (`POST /iatf/kpis`).
2. Monatlich/quartalsweise `actual` aktualisieren (`PATCH /iatf/kpis/{id}`).
3. KPIs mit `on_target=false` in QA-Review besprechen.

C — NCR → Korrekturmaßnahme (IATF 10.2)

1. NCR eröffnen (`POST /iatf/ncrs`), bei Major sofort eskalieren.
2. Optional `linked_item` auf betroffene Checklisten-Position setzen.
3. 8D-Problembericht anlegen (`/problems`) und `linked_problem` setzen.
4. NCR-Status bis `closed` führen; Major-NCRs blockieren Audit-Readiness.

D — Audit-Vorbereitung

1. `audit_ready` und `readiness_score` in der UI prüfen.
2. Audit Cockpit (`/audit-cockpit`) für Stichproben-Artefakte nutzen.
3. Norm-Matrix (`/compliance` , Tab IATF) für Klausel→Nachweis-Übersicht.

API-Endpunkte (Schreiben über API; UI derzeit read-only)

Methode	Pfad	Zweck
GET	`/iatf/summary`	Compliance-Grad
GET	`/iatf/clauses`	Klauselbaum-Rollup (Heatmap)
GET	`/iatf/audit-readiness`	Audit-Bereitschaft
POST/GET/PATCH/DELETE	`/iatf/checklist` …	Checklisten-CRUD
POST/GET/PATCH/DELETE	`/iatf/kpis` …	KPI-CRUD
POST/GET/PATCH/DELETE	`/iatf/ncrs` …	NCR-CRUD

Vollständige Verträge: `interface-control-document.md` .

Tipps

- Offene **Major-NCRs** → praktisch nie audit-ready; zuerst schließen oder enthalten.
- Serienfertigung (IATF 8.5) ist in HCQ **nicht anwendbar** — Positionen dort als `not_applicable` markieren oder weglassen.
- Norm-Matrix (`/compliance`) ergänzt IATF um ASPICE, ISO 26262 und VDA 6.3 (SWREQ-102 / SWREQ-190).
- Jede Schreiboperation erzeugt einen Audit-Trail-Eintrag (`GET /audit`).

---

25. Requirements as Code (Kurz)

Zielgruppe: PE, QM, Entwickler · **Prozess:** SWE.1 · **Traceability:** `SWREQ-215` … `219`

Requirements as Code (RaC) ergänzt die Traceability-YAML um **maschinenlesbare Validierung** — primär für Entwickler und Quality Gates, **nicht** als vollständige UI-Funktion.

Was	Wo
Single Point of Truth	`docs/traceability/traceability.yaml`
Acceptance Criteria (optional)	`docs/requirements/swreq/SWREQ-NNN.yaml`
Prozessleitfaden	`docs/process/requirements-as-code.md`
Design	`ADR-0034`

Validierung (CLI)

```
hcq-req validate          # Exit 0 = OK, 1 = Fehler
hcq-req validate --json    # für CI/Automation
```

Gate `rac-validate` in `scripts/run_gates.ps1`.

API (read-only, laufende Instanz)

```
GET /requirements-as-code/status
```

Liefert `valid`, `stats`, `issues[]` — für Audit-Nachweise nutzbar.

> **Ehrlich:** RaC validiert **Repo-Artefakte**, nicht Live-Daten in der UI. DOORS-Bi-Sync ist Roadmap; Simulation nutzt CSV-Import (§2.7).

---

26. Sicherheit, Zero Trust & Air-Gap

Design: `ADR-0017` · Matrix: `docs/security/zero-trust-compliance-matrix.md`

Zero Trust (Kurz)

Prinzip	Umsetzung in HCQ
Default-Deny	Jede API-Anfrage authentifiziert/autorisiert (`401` / `403`)
Least Privilege	RBAC-Matrix (§1, `access-control-policy.md`)
MFA (optional)	TOTP + Backup-Codes (§2.4)
Security Events	SIEM-Export JSON/JSONL/CEF (§15)
`security_posture`	`GET /health` zeigt TLS, MFA, Secrets-Backend (lokal oft aus)

> **Ehrlich:** Lokal/Development laufen TLS, Vault und mTLS typischerweise **aus**. Enterprise-PKI/HSM sind Roadmap — konfigurierbar, nicht vorinstalliert.

Air-Gap / USB-Bundle

Thema	Dokument
Offline-Bundle bauen	`docs/deployment/offline-install.md`
Kundenlieferung	`docs/deployment/LIEFERUNG-FIRMSERVER.md`
USB IT-Anleitung	`docs/deployment/USB-STICK-FIRMSERVER.md`
Deployment-Topologien	`docs/architecture/deployment-airgap.md`

Das Bundle enthält **keine Secrets** — `JWT_SECRET` und Admin-Passwort werden im Silo gesetzt.

UI-Sichtbarkeit Security

- **Security Events:** nur `ADMIN` und `QM` (§15).
- **Profil & Sicherheit:** MFA/Passkey für alle Rollen (Self-Service).

---

27. Glossar

Abkürzung	Bedeutung
SWREQ	Software Requirement (`SWREQ-###`)
SYS	System Requirement
ARC	Architecture Component
IF	Interface (Schnittstelle)
DD	Detailed Design Unit
TC	Test Case
TSPEC	Test Specification
TR	Test Run
PR	Problem Report
CR	Change Request
CI	Configuration Item
BL	Baseline
DFMEA	Design Failure Mode and Effects Analysis
L/M/H	Low / Medium / High — **maßgebliche** DFMEA-Bewertung (nicht RPN)
RPN/RPZ	Risk Priority Number — **in HCQ nicht verwendet** (ADR-0026)
HARA	Hazard Analysis and Risk Assessment
ASIL	Automotive Safety Integrity Level (QM, A, B, C, D)
S/O/D	Severity / Occurrence / Detection (DFMEA, Skala 1–10)
RASIC	Responsible, Accountable, Supported, Informed, Consulted
RTM	Requirements Traceability Matrix
APQP	Advanced Product Quality Planning
PPAP	Production Part Approval Process
PSW	Part Submission Warrant
NCR	Non-Conformance Report
AIS	AI System (Registry)
MC	Model Card
DS	Dataset Specification
AIME	AI Monitoring Event
AIIA	AI Impact Assessment
RBAC	Role-Based Access Control
MFA	Multi-Factor Authentication (TOTP)
SIEM	Security Information and Event Management
DAG	Directed Acyclic Graph (Traceability)

Abkürzung	Bedeutung
ASPICE	Automotive SPICE Prozessmodell
IATF	International Automotive Task Force (16949)
RaC	Requirements as Code (YAML-Validator, `hcq-req`)
ARF	Audit Ready Framework (Kundenname für HCQ)
DT	Digital Twin (Typ-/Instanz-Twin im Kundensilo)
CRA	EU Cyber Resilience Act — hier: **Readiness-Werkzeug**, kein Konformitätsnachweis
ATL	Attestations-Ledger (append-only Monatsabschluss-Historie)
BT / BI	Bauteiltyp / Bauteilinstanz (BaC/BIT-Register)

---

28. Anhang

API-Dokumentation

Ressource	URL / Pfad
Interaktive API-Docs (Swagger)	`http://localhost:8000/docs`
ReDoc	`http://localhost:8000/redoc`
Interface Control Document	`docs/architecture/interface-control-document.md`
Zugriffskontrolle	`docs/quality/access-control-policy.md`

Support & Betrieb

Thema	Dokument
Installation / Setup	`README.md`
Frontend-Entwicklung	`frontend/README.md`
Air-Gap / Offline	`docs/deployment/offline-install.md`
Kundenlieferung Firmenserver	`docs/deployment/LIEFERUNG-FIRMSERVER.md`
USB Firmenserver (IT)	`docs/deployment/USB-STICK-FIRMSERVER.md`
USB-Stick Transfer	`docs/deployment/USB-STICK-ANLEITUNG.md`
Lieferschein-Vorlage	`docs/deployment/LIEFERSCHEIN-VORLAGE.md`
Onboarding (10 Min.)	`docs/onboarding/FIRST-10-MINUTES.md`
Demo Quickstart	`docs/demo/QUICKSTART-DEMO.md`
Requirements as Code	`docs/process/requirements-as-code.md`
Zero Trust Matrix	`docs/security/zero-trust-compliance-matrix.md`
Framework Self-Audit	`docs/quality/HCQ-SELF-AUDIT-PACKAGE.md`
Produkt-Compliance-Paket (Dogfooding)	`docs/compliance/HCQ-PRODUKT-COMPLIANCE-PAKET.md`
On-Prem-Lizenz	`docs/deployment/ON-PREM-LIZENZ.md`
Digital Twin On-Prem	`docs/process/digital-twin-on-prem.md`
CRA-Nachweiskette	`docs/process/cra-evidence-chain.md`
SIEM-Integration	`docs/security/siem-integration-guide.md`

Häufige Stolpersteine (UI)

Problem	Ursache	Lösung
API offline	Backend nicht gestartet	Port 8000, `hcq-api` / Start-Skript
Login fehlgeschlagen	Falsche DB/Umgebung	`simulation` + `admin` / `!!(Fre1a-24)`
Leere Module	Kein Demo-Seed	`hcq-seed --profile demo --force`
Sign-off ausgegraut	RBAC/RASIC	Rolle prüfen (§14)
Material nicht gefunden	Kein Sync/Demo-Daten	§2.7, `100000002`
Integrationen Sync 403	Produktionsmodus	Nur `simulation` / `development`
Branchenschalter fehlt (Demo)	Alte Demo-Build / Auto-Login	Neu deployen; Login-Karte mit **Automotive/Robotik**, dann **Demo starten**
Robotik-Matrix leer / Roadmap	Domain noch Automotive	Kopfzeile auf **Robotik** umschalten
Einladungslink ungültig	Abgelaufen/verbraucht	Admin: neue Einladung unter `/users`
Add-on nicht sichtbar	`*_MODULE_ENABLED=false` oder Token fehlt	Env prüfen; Token unter `/profile` (§17)
CRA leer	Keine Digital-Twin-Daten	Twins anlegen (§18)
`newly_non_compliant` nach Update	Ruleset verschärft	Baseline vor Update erfassen (§19)
Zertifikat-Warnung	Ablauf in ≤90 Tagen	`/certificates` → erneuern (§20)

Qualifikationstests (Auszug SWREQ-229 / SWREQ-231)

QT	Anforderung	Testmodul	Stand
QT-229	Branchenwahl Login + Kopfzeile, Compliance-Domain	`tests/test_compliance_domain_ui.py`, `frontend/src/test/compliance-domain.test.ts`	siehe Backtest-Protokoll
QT-230	Robotik-Matrix + Diagnose-Profil	`tests/test_compliance_matrix.py`, `tests/test_diagnosis.py`	siehe Backtest-Protokoll
QT-231	Invite-Onboarding On-Prem	`tests/test_auth_invite.py`	siehe Backtest-Protokoll

Vollständiges Protokoll mit Datum und Ergebnis: `docs/quality/test-results/QT-229-231-backtest.md`

Ehrliche Grenzen (Roadmap)

Bereich	Aktueller Stand
UI-Schreibzugriff	Architektur, KI-Registry, Release-Sign-off, QA-Snapshot, Profil — sonst **API**
Material / APQP Trace	UI + API in Simulation; Live-SAP/PLM-Connector Roadmap
Requirements as Code	CLI/API-Status; **keine** dedizierte UI-Seite
Audit Cockpit PDF	Nur JSON-Export; PDF folgt
DFMEA L/M/H	Heuristische Ableitung; keine AIAG-Action-Priority-Tabelle
Compliance-Matrix	Repräsentativer Ausschnitt; kein Vollaudit-Ersatz
KI-Governance	Kein Zertifikat; Monitoring manuell
TLS/MFA/Vault	Konfigurierbar; Enterprise-PKI/HSM Roadmap
On-Prem Add-ons (DT, CRA)	Implementiert; Import-GW, SBOM, VDP **geplant**
Zertifizierung	HCQ/ARF **ist nicht zertifiziert** — Nachweise im Repo, ehrliche Statusführung

---

29. Berater: API-Anbindung beim Kunden

Zielgruppe: HCQ-Berater (Audit-Sprint, Diagnose, Vor-Ort-Integration) · **Anforderung:** `SWREQ-224`

Das Anwendungshandbuch beschreibt primär die **Web-UI**. Für die **technische Anbindung beim Kunden** (Exporte, REST-Import, Service-Account, Sprint-Checkliste) gibt es ein eigenes Berater-Dokument:

`API-ANBINDUNG-KUNDE.md`

Thema	Wo im Berater-Handbuch
JWT-Login & Service-User	§4–5
CSV/Simulation (DOORS/SAP/PLM)	§6
REST direkt (Requirements, Tests, Trace)	§7
Mapping Kunde → HCQ	§8
ALM: Jira / Jama / Codebeamer / Windchill	§8.1–8.4 (`SWREQ-227`)
Referenz-Skript Jira → HCQ	§15 · `scripts/sync-jira-to-hcq.py`
Checkliste Tag 1–3	§10
PowerShell/curl	§11
Live-Connectoren Roadmap	§13

> **Kurz:** In **Produktion** kein `/integrations/sync` — Daten per **REST** oder CSV-Skript einspielen. OpenAPI unter `/docs` am Kunden-Server ist Vertragsreferenz.

Dokumentenhistorie

Version	Datum	Änderung
0.3.0	2026-07-07	On-Prem-Lizenz, Digital Twin, CRA, Zertifikatsregister, CRA-Nachweiskette (SWREQ-251..264)
0.2.0	2026-06-18	Produktname ARF, Material/APQP (SWREQ-096), RaC, Zero Trust, Auditor-Workflows, Simulation-Zugang
0.2.2	2026-06-30	ALM-Anbindung Jira/Jama/Codebeamer/Windchill, Jira-Referenz-Skript (`SWREQ-227` , ADR-0036)
0.2.1	2026-06-30	Verweis Berater-Handbuch API-Anbindung (`SWREQ-224`)
0.1.0	2026-06-17	Erstfassung Anwendungshandbuch (`SWREQ-210`)

Full English translation: `USER-MANUAL-EN.md`