Funktionale Anforderungen – F:

F 1: Stay-in-sync-Service (Grundlage)

F 1.1: Konfiguration & Verwaltbarkeit
F 1.1.1: Die initiale Konfiguration soll über eine .config-Datei eingespielt werden.
F 1.1.2: Eine REST-API soll das Hinzufügen oder Löschen von Konfigurationen erlauben.
F 1.1.2.1: Eine Startoption des Services soll steuern, ob erweiterte Konfigurationen in der .config-Datei persistiert werden.

F 1.2: Schnittstellen- & Systemunterstützung
F 1.2.1: Zu unterstützende Datenformate: JSON und JSON-LD.
F 1.2.2: HTTP-REST-API: GET, POST, PUT, DELETE, QUERY.
F 1.2.3: Protokolle: HTTP, HTTPS, WebSocket-Streams.
F 1.2.4: Selbst signierte Zertifikate sollen unterstützt werden.
F 1.2.5: Authentifizierungsmethoden: No-Auth, Basic-Auth, Bearer-Token, Single Sign-On, OAuth 2.0, Catena-X-Auth (Dynamic Attribute Provisioning Service, Decentralized Claims Protocol).
F 1.2.5.1: Die gespeicherten Authentifizierungsinformationen müssen durch ein Sicherheitskonzept vor unbefugtem Zugriff geschützt werden.
F 1.2.6: Dedizierte Unterstützung für die API-Struktur von EDC- und AAS-Schnittstellen.
F 1.2.6.1: Integration der HTTP-, S3- und Azure-Blob-Storage-API.

F 1.3: Synchronisationslogik
F 1.3.1: Regelbasierte Synchronisierung mehrerer Quellsysteme in ein Zielsystem.
F 1.3.2: Zeitgesteuerte Aktualisierung mit einem Minimum von 1 ms.
F 1.3.3: Bedingungsbasierte Aktualisierung, wenn definierte Änderungsbedingungen erfüllt werden.
F 1.3.3.1: ODER-Bedingung: Mindestens eine Variable ändert sich gegenüber dem vorherigen Synchronisationsstand.
F 1.3.3.2: UND-Bedingung: Alle betrachteten Variablen ändern sich.
F 1.3.3.3: Logische Bedingungen mit booleschen und mathematischen Operatoren.
F 1.3.3.4: (Optional) Zeitlich begrenzte logische Bedingungen (z. B. alle Parameter ändern sich innerhalb von 30 s).
F 1.3.3.5: (Optional) Verifizierung für den Snapshot, dass die Datenkonstistenz gewährleisten über Regeln

F 1.3.4: Die Auslösung der Aktualisierung soll Snapshots der Quellsysteme nutzen, um skriptbasiert Variablen ins Zielsystem zu schreiben.

F 1.4: Skript-Engine & Transformation
F 1.4.1: Skripte sollen TypeScript, Python oder C-Scripts unterstützen (Operationen, Transformationen, Bibliotheken).
F 1.4.1.1: (Optional) Paket-Management-Funktionen zum Nachladen externer Bibliotheken.
F 1.4.1.2: (Optional) Wiederverwendbare Transformations-Bibliotheken in Kombination mit dem Konfigurator.
F 1.4.2: Skripte werden als Teil der Konfiguration eingespielt.
F 1.4.3: Sicherheitskonzepte müssen Code-Injection über Variablen der Quellsysteme verhindern.

F 1.5: Fehlertoleranz & Logging
F 1.5.1: Tritt in einer Anfrage-Sequenz ein Fehler auf, muss eine Fehlerstrategie umgesetzt werden.
F 1.5.1.1: Jeder Fehler muss mindestens geloggt werden.
F 1.5.1.2: (Optional) Eine Rollback-Strategie soll ausgeführt werden.
F 1.5.2: Logging nach typischen Log-Leveln und zentrale Zusammenführung über das Monitoring-Werkzeug.

F 1.6: Betriebsmodi & Skalierung
F 1.6.1: Die Anwendung muss horizontal skalierbar sein.
F 1.6.2: Ein Konzept zur Vermeidung kaskadierender Effekte und Trigger-Schleifen wird implementiert.
F 1.6.3: Der Service bietet einen Trocken-/Simulationsmodus, der Aktionen ausführt, aber nicht ins Zielsystem schreibt.

F 2: Monitoring & Deployment
F 2.1: Deployment-Skripte ermöglichen Stand-alone-, Docker- und Kubernetes-Betrieb.
F 2.2: Die Monitoring-App übernimmt Umgebungskonfigurationen aus dem Konfigurator.
F 2.3: Aggregation von Logging-Informationen.
F 2.4: UI in Graph-Form mit Darstellung der Datenflüsse und Konfigurationen.
F 2.4.1: Metriken an den Knoten (Sync-Diensten). → Request-Anzahl, Systemauslastung, Last pro Skript / Ausführungszeiten / Quell & Zielsystem
F 2.4.2: Einzelansicht der Service-Logs.

F 2.5: Aggregierte Ansicht der Service-Logs für das gesamte System.
F 2.5.1: Filter nach Ziel-/Quellsystem oder Services.
F 2.5.2: (optional) Skaliert laufende Skripte → Loggs zusammenführen

F 3: Konfigurator
F 3.1: EDC-Verwaltung
F 3.1.1: Verwaltung von Business Partner Networks & Identitäten (Tractus-X, Catena-X).
F 3.1.1.1: (Optional) Auslesen bestehender Netzwerke aus einem Identity Provider.
F 3.1.2: EDC-Instanzen anlegen, aktualisieren, löschen.
F 3.1.2.1: (Optional) Direktes Auslesen aus der aktuellen EDC-Konfiguration.
F 3.1.3: Unterstützung zum Deployment eines eigenen EDC.
F 3.1.4: Freizugebende Variablen aus Ziel-/Quellsystemen definieren.
F 3.1.4.1: Zuweisung zu mehreren Identitäten.
F 3.1.4.2: Konfliktfreie Aktualisierung des EDC-Catalogs.

F 3.2: AAS-Verwaltung
F 3.2.1: AAS-Instanzen anlegen, aktualisieren, löschen.
F 3.2.1.1: Bereitstellung einer eigenen AAS als Zielsystem.
F 3.2.2: Einlesen der von einer AAS angebotenen Datentypen.
F 3.2.2.1: Ergänzung fehlender Spezifikationen für Zielsysteme.

F 3.3: IDE-Unterstützung
F 3.3.1: Autovervollständigung für konfigurierte Datentypen und Schnittstellen.
F 3.3.2: Unterstützung der Sync-Service-Konfiguration.
F 3.3.3: (Optional) Bibliotheksverwaltung für Sprach-abhängige Dependencies.

F 3.4: Umgebungskonfiguration
F 3.4.1: Graph-basiertes Verbinden von Sync-Services, EDC, AAS und weiteren APIs.
F 3.4.2: Aufspielen der Konfiguration auf alle relevanten Systeme.
F 3.4.3: Testumgebungsauführung (Beispiel Daten für API-Request/Änderungen in der Request/Simuliertes Zielsystem)

F 3.5: Verwaltung von Quell- & Zielsystemen
F 3.5.1: Auslesen verfügbarer Schnittstellenpfade und Datentypen (z. B. über OpenAPI).
F 3.5.1.1: (Optional) Analyse von Beispielanfragen.
F 3.5.1.2: Unterstützung von AAS oder REST-APIs mit JSON/JSON-LD, direkt oder über EDC.

F 3.5.1.3: Dedizierte Datentyp-Unterstützung bei AAS-Systemen.

Nicht-funktionale Anforderungen – NF:

NF 1: Die Lizenz muss eine gängige Open-Source-Lizenz sein (EPL-2.0, Apache-2.0, BSD-3-Clause, MIT).
NF 2: ≥ 90 % Testabdeckung durch Unit-, Integrations- und Systemtests.
NF 3: Entwickler- und Benutzerdokumentationen in Englisch (≥ 90 % Deckungsgrad).
NF 4: Keine “Critical”/“Blocker” und ≤ 5 % “Major”-Issues in statischen Analysen; technische Schulden < 5 %.
NF 5: Containerisierte Anwendung (OCI-Images) für x86-64 und ARM64.
NF 6: Horizontale Skalierung; bei fünffacher Laststeigerung ≥ 90 % Performanceziele ohne Code-Änderung.
NF 7: Gut dokumentierte Repositories:

Dokumentation der Applikation und des Repository-Aufbaus.

Entwickler-Guide für den Projekteinstieg.

Startup- & Installations-Guides.

Sauber kommentierter Quellcode gemäß Styleguide, ohne Code-Smells und mit Design-Patterns.