SkillPilot Konzeptkompendium · Band III

Personalisierung, Lernlaufzeit und Coach-Architektur

Vier-Ebenen-Personalisierung, Lern- und Synchronisationsabläufe, portabler

MCP-Host-Rahmen, ChatClient/Backend-Vertrag und Plugin-Lebenszyklus

Geltungsbereich dieses Kompendiums

Dieser Band vereinigt die zehn normativen Laufzeit- und Coach-Spezifikationen von Skill Pilot: die vier Ebenen der Personalisierung, den operativen Lernablauf samt binärem Mastery-Vertrag, Browser-SRS-Synchronisation und signierten Import/Export, die langfristige Architektur für austauschbare MCP-Hosts, den providerneutralen Kommunikationsvertrag zwischen ChatClient und Backend, die Produkt- und Adaptertopologie, die strikte Trennung von OAuth-Appbindung und 24-Stunden-Lernsession, die Golden Journeys der Verhaltensintegration sowie den Versionierungs- und Lebenszyklusplan inklusive der Entscheidung vom 19. Oktober 2026 für das dauerhafte Plugin `skillpilot-coach`.

Stand: 11. November 2026 · Lizenz: Apache-2.0 (technische Architekturspezifikation; siehe LICENSING.md).

Lokale Kandidaten und Dokumentationsstände ersetzen keine reale Host-Abnahme oder Portal-Veröffentlichung.

Inhaltsverzeichnis

1 Ebenen der Personalisierung 4
1.1 Vor Ebene 1: Katalognavigation . . . 4
1.2 Ebene 1: Basis-Curriculum . . . 4
1.3 Ebene 2: Persönlicher Lehrplan (Personal Curriculum) . . . 4
1.4 Ebene 3: Lernfokus und aktives Lernziel . . . 5
1.5 Ebene 4: Mastery und orthogonale Zeitachse . . . 5

2 Operativer Lernablauf 6
2.1 Datengrundlage: Landschaften, Struktur und Filter . . . 6
2.2 Lebenszyklus-Schritte 1–3: Basis-Curriculum, Personal Curriculum und Fokus . . . 6
2.3 Schritt 4: Frontier-Berechnung und Auswertungsmodi . . . 6
2.4 Schritt 5: Mastery, Aggregation und Rückmeldung . . . 7

3 Client-State-Synchronisation für Browser-SRS-Fortschritt 8
3.1 Endpunkt und Authentifizierungsgrenze . . . 8
3.2 Request- und Response-Struktur . . . 8
3.3 Speicherung, Datenschutz und Client-Verhalten . . . 9

4 Import- und Export-Workflow 10
4.1 Export, Signaturprüfung und Import des Lernstands . . . 10
4.2 Lokale Klassendateien von Lehrkräften . . . 10

5 Architekturkonzept: SkillPilot mit austauschbarem MCP Host 12
5.1 Teil A — Stabile Architektur: Drei-Ebenen-Trennung und Kerngrenzen . . . 12
5.2 Zustandsgrenzen, Betriebswege, Medien und Abrechnung . . . 13
5.3 Stufenplan und verbindliche Host-Abnahmetests . . . 14
5.4 Teil B — Datierter Marktanhang (Stand 19. August 2026) . . . 14

6 Kommunikationsvertrag zwischen ChatClient und SkillPilot-Backend 15
6.1 Kernsatz, Verantwortungsmatrix und minimaler Turn-Vertrag . . . 15
6.2 Werkzeuge und Eingaben: semantisch klein, frei von Chat-Freitexten . . . 16
6.3 Ergebnisse, opake Capabilities und Referenz Verified Recall . . . 17
6.4 Auswahl, Fehler, Datenschutz, Anti-Patterns und 11-Punkte-Abnahme . . . 17

7 SkillPilot-Lerncoach: Produkt- und Providerarchitektur 19
7.1 Produktentscheidung und mehrsprachige Plugin-Struktur . . . 19
7.2 Systemtopologie und Schichtenverantwortung . . . 19
7.3 Web-first-Einstieg, sichere Projektion und UI-Oberfläche . . . 20
7.4 Bewertungsgrenze, weitere Provider und 8 Freigabegates . . . 20

8 OpenAI-MCP-App: OAuth- und 24h-Lernsession-Bindung 22
8.1 Architekturentscheidung: Zwei getrennte Berechtigungen . . . 22
8.2 Verbindlicher Startablauf, Erneuerung und Diagnose-TTL . . . 23
8.3 MCP-Validierungsreihenfolge, Aktionshorizont und Verbote . . . 23
8.4 Persistenzschema, OAuth-/mTLS-Härtung und Fehlerverhalten . . . 24
8.5 Sicherheitsinvarianten und automatisierte Abnahmekriterien . . . 25

9 Verhaltensintegration des MCP-Lerncoaches 27
9.1 Erfolgsbild und nicht verhandelbare Invarianten . . . 27
9.2 Allgemeiner 10-stufiger Entscheidungszyklus, Scope vs. Fokus und Fehlerbehandlung 27
9.3 Wirksame Laufzeitorte und verbindliche Verhaltensregeln . . . 28

9.4 Golden Journeys (`GJ-01` bis `GJ-11`) 30
9.5 Testpyramide, Qualitätsziele, Testprotokoll und Definition of Done 31

10 SkillPilot: Versionierungs- und Lebenszyklusplan für das OpenAI-Plugin 33
10.1 Entscheidung vom 19. Oktober 2026: ein dauerhaftes neutrales Plugin 33
10.2 Unabhängige Versionsebenen und SemVer-Klassifikation 34
10.3 Öffentliche Vertragsadapter, Envelopes und dedizierte MCP-Apps-UIs 35
10.4 Kanonisches `contract-line/v1`-Schema, CI-Gates, Observability und Linienhistorie 36

1 Ebenen der Personalisierung

Dieses Kapitel beschreibt, wie SkillPilot vom veröffentlichten Curriculum-Katalog bis zur individuellen atomaren Beherrschung eingrenzt. Die vorgeschaltete Katalognavigation liegt vor den vier Personalisierungsebenen und wird niemals als Lernendenzustand gespeichert.

Diese vier Ebenen sind strikt von Lerneinheiten (`LearningUnit`) zu unterscheiden — rekursiv zusammengesetzten didaktischen Einheiten mit Lernversprechen und abgeleitetem Abschluss, deren Verortung zum aufgelösten Bildungskontext gehört und keine fünfte Personalisierungsebene bildet.

1.1 Vor Ebene 1: Katalognavigation

Lernende wählen zunächst einen Katalogbereich wie **Schule**, **Hochschule** oder **Sprachen und Weiterbildung**, um ein passendes Wurzel-Curriculum zu finden. Dieser Schritt dient ausschließlich der Orientierung im Katalog.

1.2 Ebene 1: Basis-Curriculum

Gewählt wird genau ein veröffentlichtes Wurzel-Curriculum, das den vollständigen Zielraum bereitstellt, ohne kanonische Lernziele zu kopieren oder umzubenennen (z. B. `Gymnasium (DE)`, ein Studiengangskatalog oder eine CEFR-Sprach-Skill-Landschaft).

Nur in `curricula/curriculum_manifest.json` deklarierte Wurzel-Curricula sind in der Oberfläche wählbar und für Champion-Registrierungen zulässig:

```
{
  "curricula: [
    { id: ä0e13c56-c25f-4742-9272-3a1a603ee52e", "title: "Gymnasium (DE)" }
  ]
}
```

Jede Manifest-ID muss unter `curricula/` existieren, darf keine Teil- oder Modullandschaft sein und wird in CI gegen die berechnete Wurzelmenge geprüft.

1.3 Ebene 2: Persönlicher Lehrplan (Personal Curriculum)

Der Persönliche Lehrplan ist der verbindliche, längerfristige Lernumfang innerhalb des Basis-Curriculums — er bestimmt die geltende curriculare Sicht und nicht den kurzfristigen Arbeitsfokus. Für deutsche Schulcurricula gilt die semantische Reihenfolge:

1. **Bundesland / Jurisdiktion** (einschließlich expliziter kanonischer Sicht, sofern verfügbar);
2. **Dauermodell** (z. B. G8 oder G9, sofern landesspezifisch relevant);
3. **Stufe** (Sekundarstufe I, Sekundarstufe II oder ausdrücklich beide);
4. **Ausgewählte Fächer**;
5. **Kursprofil je gewähltem Fach** (sofern einschlägig).

Das Kursprofil hängt am einzelnen Fach: Mathematik `LK` und Physik `GK` können parallel gelten, `LK` impliziert niemals automatisch Sekundarstufe II, und eine fehlende Stufenangabe wird niemals stillschweigend stufenübergreifend gedeutet. Ebene 2 löst die geltende Kompositionsansicht auf und begrenzt Fokus, Frontier, Fortschritt und Abschluss; Änderungen revalidieren Fokus und `activeGoal`, ohne stabile Ziel-IDs oder globale Mastery zu löschen.

Einheitlicher First-Party-Editorvertrag: Ebene 2 wird ausschließlich im First-Party-Web GUI von SkillPilot gepflegt (vor dem Lernstart auf dem Startbildschirm sowie später im Cockpit) über gemeinsame Backend-Optionen, Validierung und Speicherung. Der OpenAI-ChatClient fragt Ebene 2 weder ab noch verändert er sie; ist sie unvollständig, stoppt er das Fachcoaching und verweist auf das WebGUI.

1.4 Ebene 3: Lernfokus und aktives Lernziel

Ebene 3 ist bewusst leichtgewichtig veränderbar und besteht aus zwei Teilen:

- **Ebene 3a — Fokus:** eine oder mehrere Fokuswurzeln, die den aktuellen Lernkorridor wählen (z. B. Jahrgangsstufe 7, Q1 oder Analysis; Programmeinheiten sind selbst keine kanonischen Lernziele).
- **Ebene 3b — Aktives atomares Ziel:** höchstens ein konkretes, überprüfbares atomares Lernziel als unmittelbares Arbeitsziel.

Bei einer Fokusweitung veröffentlicht das Backend entlang des sichtbaren Vorfahrenpfads zur Wurzel nur gültige breitere Vorfahren, die noch offene `target`-Ziele ergänzen (nächstgelegener Vorfahr zuerst, unter Erhalt unabhängiger Mehrfachwurzeln in `goalIds`). Ein automatischer Vorschlag erfolgt nur nach tatsächlichem Abschluss des aktuellen Fokus (nicht bei bloß blockierter Frontier) und erfordert die Zustimmung der lernenden Person.

Nach der Korrektur vom 4. November 2026 (Abschlusskonzept) schreibt `requires` allein keine Voraussetzungs-Mastery; bestandene Anwendungsprüfungen begründen global wirksame Anerkennung gewöhnlicher Ziele, die beim Fokusweiten bestehen bleibt.

1.5 Ebene 4: Mastery und orthogonale Zeitachse

Mastery ist das einzige dauerhafte Ergebnis je stabiler atomarer Ziel-UUID — **gemeistert oder nicht gemeistert** — und gilt global über Kompositionsansichten sowie Ebene-2/3-Wechsel hinweg ohne Teilbeherrschungsstufen. Effektive Beherrschung $M_{\text{eff}} = M \cup W$ vereint gespeicherte explizite Mastery M und prüfungsbasierte Anerkennung W (Prüfungs- und Lernkartenziele bleiben explizit; Laufzeitmigration offen, siehe 2.4). Prozentwerte auf Clustern bezeichnen den gewichteten Anteil effektiv beherrschter Atomziele, niemals Teilmastery eines Einzelziels.

Orthogonale Zeitachse: Ein datierter Kurs- oder Klausurplan ist kein fünftes Personalisierungslevel, sondern eine Ausführungsplanung für das **Wann** innerhalb von Ebene 2 (siehe Einheitlicher Lernplanstatus und Zeitachsenkonzept).

2 Operativer Lernablauf

Dieses Kapitel beschreibt den operativen Laufzeitablauf von statischen Skill-Landschaften unter `curricula/` bis zur personalisierten Navigation und binären Mastery-Fortschreibung.

Im Backend ist der Ablauf providerneutral. Für ChatGPT (8) authentisiert ein registrierter OAuth-Client die Integration, während jeder ausdrückliche Klick auf **Lernen starten** eine frische, absolut nach 24 Stunden ablaufende `learningSessionId` erzeugt, in die Startnachricht einfügt und unverändert an jeden fachlichen MCP-Aufruf übergibt; die permanente SkillPilot-ID verbleibt ausschließlich im Backend. Der Visible-Session-Flow dient nur als isolierter Rollback-Pfad, und der Claude-Adapter nutzt dieselbe sichere Zustandsprojektion vorbehaltlich vollständiger End-to-End-Abnahme. Die gemeinsame Grenze regelt 6.

2.1 Datengrundlage: Landschaften, Struktur und Filter

Jede Skill-Landschaft unter `curricula/` bildet einen DAG mit `contains` (Hierarchie/Komposition) und `requires` (didaktische Voraussetzungen). Während der laufenden Clusterkanten-Migration leitet die Laufzeit **effektive Voraussetzungen** aus direkten `requires`-Kanten sowie über `contains`-Vorfahren vererbten `requires`-Kanten ab; Personal-Curriculum-Filter grenzen die sichtbare Ansicht ein, während für strikte Prüfungen weiterhin der strukturelle Gesamtgraph zur Verfügung steht.

2.2 Lebenszyklus-Schritte 1–3: Basis-Curriculum, Personal Curriculum und Fokus

1. **Basis-Curriculum (Ebene 1):** Nach zustandsfreier Katalognavigation wählt die lernende Person ein veröffentlichtes Wurzel-Curriculum als vollständigen Zielraum.
2. **Verbindliches Personal Curriculum (Ebene 2):** Ausschließlich im First-Party-WebGUI konfiguriert (Bundesland/kanonische Sicht → Dauermodell → Stufe → Fächer → Kursprofil je Fach) als Kandidatenraum für Fokus, Frontier, Fortschritt und lokale Lehrkraft-Kursplanung (unabhängig vom veränderlichen Ebene-3-Fokus).
3. **Fokus und aktives Ziel (Ebene 3):** - **Ebene 3a (Fokuswurzeln):** Sobald Ebene 2 vollständig ist, ist der Fokus nie leer (deterministischer Standard auf der ersten sichtbaren Wurzel; Mehrfachwurzeln für kombinierte Backend-/Coach-Korridore möglich). - **Ebene 3b (Aktives Atomziel):** Höchstens ein atomares Lernziel aktiv in UI und Coach-Schleife. - **Plangesteuerte Progression:** Die bestätigte Planübernahme ersetzt einbezogene Fachpläne atomar, aktiviert den Planmodus (unterdrückt den generischen Autopilot) und wählt das erste fällige Frontier-Ziel eines offenen Periodenpensums; explizite Fachwechsel parken offene Ziele ohne Mastery-Änderung, und Übergaben nach Zielabschluss folgen deterministisch offenen Fachpensen ohne jemals `requires` zu umgehen oder Ersatzziele zu erfinden (siehe Einheitlicher Lernplanstatus sowie historisch Tagesfortschritt). - **Fokusweitung:** Bietet vom Backend publizierte breitere Vorfahren entlang des sichtbaren Wurzelpfads erst nach echtem Fokusabschluss und ausdrücklicher Zustimmung an.

2.3 Schritt 4: Frontier-Berechnung und Auswertungsmodi

Ein atomares Ziel liegt genau dann auf der **Frontier**, wenn es selbst noch nicht beherrscht ist und sämtliche effektiven Voraussetzungen gemäß ihren Zieltyp-Verträgen erfüllt sind. Nach der Korrektur vom 4. November 2026 (Abschlusskonzept; Laufzeitanpassung offen) erfüllt effektive Beherrschung $M_{\text{eff}} = M \cup W$ Voraussetzungen ohne synthetische Mastery-Zeilen, während eine

gewöhnliche B-Mastery in $Q \to B \to A$ niemals A anerkennt, solange die lokale Prüfung Q offen ist.

- **Optimistischer Modus:** Prüft Voraussetzungen innerhalb des aktiven Fokus (berechnet transitive Erreichbarkeit vor Zielmaskierung, damit ausgeblendete Zwischenknoten keine Pfade kappen) und erlaubt den direkten Einstieg im gewählten Bereich.
- **Strikter Modus:** Wählt Kandidaten aus dem gefilterten Bereich, prüft Voraussetzungen jedoch global, um fehlende Grundlagen außerhalb des Filters offenzulegen.

Wünscht die lernende Person ein Nicht-Frontier-Ziel, ermittelt die Laufzeit die fehlenden effektiven Voraussetzungen und leitet gezielt dorthin.

2.4 Schritt 5: Mastery, Aggregation und Rückmeldung

Jedes atomare Ziel besitzt auf seiner stabilen UUID genau ein gespeichertes Ergebnis: **gemeistert oder nicht gemeistert** (`0.0` oder `1.0`; Zwischenwerte sind kein dritter Lernstand).

- **Globale Geltung und Aggregation:** Mastery bleibt über Ansichts-, Ebene-2- und Ebene-3-Wechsel hinweg global erhalten. Cluster- und Bereichsprozentwerte zeigen den gewichteten Anteil eindeutig beherrschter Atomziele (60 % bedeutet z. B. 6 von 10 Zielen beherrscht, niemals 60 % eines Einzelziels).
- **Gedächtnis-/SRS-Ziele:** Leiten ihr binäres Zielergebnis aus dem Kartenstatus ab: Normales Karteikartenlernen im Cockpit oder in der dedizierten MCP-App-Komponente (*Noch nicht gewusst* / *Gewusst* mit Vor-/Zurück-Navigation) aktualisiert nur den Wiederholungsplan, während **Verified Recall** ein separater hilfefreier Coach-Batchtest ist. Ein Gedächtnisziel gilt erst als gemeistert, wenn alle Pflichtkarten verifiziert und aktuell nicht fällig sind (siehe Knotentypen). Orientierungsziele speichern lediglich den abgeschlossenen Einstiegsdialog ohne Fachkompetenzanspruch.
- **Implementierungsstand (geprüft am 30. Oktober 2026):** `LearnerService` akzeptiert bei neuen allgemeinen Mastery-Schreibzugriffen nur `0.0` oder `1.0`, und `GoalCard.tsx` bietet binäre Steuerelemente; `Mastery.java` behält `double` samt Leseschwelle `>= 0.9` für Bestandszeilen. Browser-SRS-Synchronisation kann keinen Verified Recall erzeugen, und signierte Importe bewahren verifizierte Kartenergebnisse (siehe Completion-Pilot). Bei jeder Mastery-Änderung werden freigeschaltete Ziele, Frontier und Fortschrittsübersichten unmittelbar neu berechnet.

3 Client-State-Synchronisation für Browser-SRS-Fortschritt

Dieses Kapitel definiert den UI-seitigen Backendvertrag zur Synchronisation des lokalen Karteikarten-Wiederholungsstands (Spaced Repetition System, SRS). **Verified Recall** aktualisiert verifizierte Kartenergebnisse ausschließlich über seinen eigenen, Capability-gebundenen atomaren MCP-Batch-Use-Case (6.3), niemals über diesen Browser-Endpunkt.

3.1 Endpunkt und Authentifizierungsgrenze

```
GET /api/ui/learners/{skillpilotId}/client-state/{nodeId}
PUT /api/ui/learners/{skillpilotId}/client-state/{nodeId}
```

- **First-Party-Browserpfad:** Nutzt die obigen UI-Endpunkte mit der pseudonymen `skillpilotId`, um den SRS-Planungsstand je Gedächtnisknoten (`nodeId`) periodisch, manuell sowie für Export/Import-Wiederherstellung frei von personenbezogenen Daten zu sichern.
- **MCP-Coach-Pfad:** Authentisiert die App per OAuth und adressiert Lernende ausschließlich über die aktive 24-Stunden-`learningSessionId` (8); die `skillpilotId` ist niemals MCP-Toolargument oder -ergebnis. Frühere kartenweise Visible-Session-Routen gehören ausschließlich zum isolierten Visible-Session-Rollback.

3.2 Request- und Response-Struktur

`PUT` überträgt `Content-Type: application/json` mit der Karten-Map eines Gedächtnisknotens (`nodeId`):

```
{
  üpdatedAt: "2026-02-02T19:30:00.000Z",
  ßrsState: {
    "math_analysis_c01: {
      id: "math_analysis_c01",
      interval: 3,
      ëaseFactor: 2.4,
      "repetitions: 2,
      "nextReview: 1706892870000,
      "lastReviewed: 1706806470000,
      "verifiedRecall: {
        ßtatus: "passed",
        ättempts: 1,
        "failures: 0,
        "lastTestedAt: "2026-02-02T19:28:00.000Z",
        "passedAt: "2026-02-02T19:28:00.000Z"
      }
    }
  }
}
```

- `srsState` speichert die SRS-Felder (`interval`, `ef`/`easeFactor`, `repetition`/`repetitions`, `nextReview`, optional `lastReviewed`) sowie abwärtskompatible `verifiedRecall`-Metadaten (Browser-Sync kann keinen neuen Verified-Recall-Erfolg fälschen; siehe 2.4).
- `GET` liefert `200 OK` mit `{ üpdatedAt: "...", ßrsState: { ... } }` (bzw. `404 Not Found`, worauf der Client lokal weiterarbeitet).
- `PUT` bestätigt `200 OK` mit `{ ßtatus: ök", ßavedAt: "...", ßtoredKeys: 42 }`.

3.3 Speicherung, Datenschutz und Client-Verhalten

- **Persistenz:** Speicherung als JSON-Datensatz pro `(learner, nodeId)` (z. B. `learner_client_state`) nach `updatedAt` ohne personenbezogene Angaben (keine Namen oder E-Mail-Adressen).
- **Client-Auslöser:** Automatische Synchronisation nach jeweils **20** bearbeiteten Karten sowie über manuelles **Speichern**; schlägt der Abgleich fehl, zeigt die UI den Fehler an und führt das lokale Üben unterbrechungsfrei fort.

4 Import- und Export-Workflow

Dieses Kapitel spezifiziert die Portabilitätsabläufe für den Umzug pseudonymer Lernstände zwischen SkillPilot-Konten bzw. -Instanzen sowie für die clientseitige Verschlüsselung lokaler Lehrkraft-Klassendateien.

4.1 Export, Signaturprüfung und Import des Lernstands

Der Lernstands-Import/Export gewährleistet **manipulationsgeschützte Datenportabilität**:

- **Enthalten:** gewähltes Curriculum, aktivierte Module/Personalisierung, Mastery-Zustand, geplante Ziele, Provenienz (`copySources`) und clientseitiger SRS-Zustand.
- **Ausgeschlossen & garantiert:** keinerlei Klarnamen/PII; ein Export verändert das Quellprofil nie, und beim Import behält das Zielprofil seine eigene `skillpilotId`.

1. **Export:** Das Backend erstellt ein per **HMAC-SHA256** signiertes `serverExport`-Paket (berechnet über geparste Werte, sodass Whitespace/Schlüsselreihenfolge unkritisch sind, jede Wertänderung jedoch die Signatur bricht), und das Frontend bündelt es zusammen mit `clientData.srsState` in einem versionierten `"2.0"`-JSON-Umschlag.
2. **Import:** Das Backend prüft die HMAC-SHA256-Signatur, übernimmt die Curriculum-Einstellungen, aktualisiert die binären Beherrschungsergebnisse importierter Ziele (unberührte Ziele bleiben erhalten; historische Zwischenwerte sind reine Speicherkompatibilität, siehe 2.4), ersetzt geplante Ziele, hängt Quell-`skillpilotId` und Zeitstempel an die `copySources`-Provenienz an und lässt den Browser `srsState` auf die Ziel-ID umschlüsseln. Nach der Produktkorrektur vom 4. November 2026 wird effektive Beherrschung aus expliziten Ergebnissen und ihren festen Autorenverträgen abgeleitet, ohne gültige Anerkennung durch Planwechsel zu entziehen oder alte Ergebnisse durch spätere Graphkanten auszuweiten.

```
{
  "version: "2.0",
  ëxportedAt: "2024-05-20T10:00:00.000Z",
  ßerverExport: {
    "data: { "learner: {}, "mastery: {}, "plannedGoals: [], "copySources: [] },
    ßignature: «hmac-sha256>"
  },
  "clientData: {
    ßrsState: { ßrs_state_USERID_GOALID: {} }
  }
}
```

4.2 Lokale Klassendateien von Lehrkräften

Lokale Trainer-Klassen bilden einen **getrennten, vollständig im Browser verschlüsselten Pfad**, da sie Klarnamen/Aliase auf permanente SkillPilot-IDs abbilden; sie sind keine Lernstandsexporte und werden vom Backend weder gesehen noch signiert:

1. Die Lehrkraft vergibt und bestätigt eine Passphrase von mindestens **15 Zeichen** (mit konsistenter Unicode-Normalisierung vor der Schlüsselableitung).
2. Eine validierte lokale `ClassSession` wird als versioniertes `skillpilot-trainer-class`-Payload serialisiert.
3. **PBKDF2-SHA-256** (`600.000` Iterationen, frischer 16-Byte-Salt) leitet einen **AES-256-GCM**-Schlüssel ab (frischer 12-Byte-IV, 128-Bit-Tag, feste Klassendatei-AAD).

4. Ausschließlich der verschlüsselte Umschlag wird unter dem neutralen Dateinamen `skillpilot-class-YYYY-MM-DD.skillpilot` heruntergeladen (ohne Klassennamen im Dateinamen).

Algorithmus, Iterationszahl, Version, Zweck, Base64-Felder, Ciphertext und entschlüsseltes Payload werden vor der Übernahme strikt geprüft: Erkannte verschlüsselte Umschläge fallen niemals auf Klartext-Parsing zurück, und falsche Passphrasen sowie manipulierte Daten erzeugen dieselbe nicht-technische Fehlermeldung. Ältere Klartextdateien (`skillpilot-class-*.json`) bleiben mit explizitem Hinweis auf erneuten verschlüsselten Export importierbar. Da entschlüsselte Klassendateien permanente SkillPilot-IDs mit vollem Lernendenzugriff wiederherstellen (siehe Lehrkraftansicht per SkillPilot-ID), sind sie wie Bearer-Secret-Container zu schützen; die Dateiverschlüsselung schützt Downloads *at rest*, nicht ein entsperrtes lokales Browserprofil.

5 Architekturkonzept: SkillPilot mit austauschbarem MCP Host

Geltungsbereich und Aufbau

Strategiekonzept für die langfristige Host- und Anbieterunabhängigkeit von SkillPilot. Die verbindlichen Laufzeitverträge des ausgelieferten Coach-Pfades stehen in 6 bis 10; bei Abweichungen gelten jene Verträge. **Teil A** definiert die stabile, normative Zielarchitektur; **Teil B** dokumentiert den datierten Marktanhang (Stand 19. August 2026, ergänzt um die nachfolgenden Repo-Entscheidungen aus 10).

5.1 Teil A — Stabile Architektur: Drei-Ebenen-Trennung und Kerngrenzen

SkillPilot besteht langfristig aus drei unabhängig austauschbaren Bereichen:

1. **SkillPilot Core** als dauerhaftes, anbieterneutrales Lernsystem,
2. einer **austauschbaren AI Chat Application bzw. einem MCP Host**,
3. einem **austauschbaren Model Provider** (bevorzugt über eine Open-Responses-kompatible Schnittstelle).

Zwei gleichrangige Architekturregeln

1. **SkillPilot Core darf weder von einem bestimmten Chat-Host noch von einem bestimmten LLM-Anbieter abhängig sein.**
2. **SkillPilot Core persistiert pro Lernendem ausschließlich strukturierten, schema-validierten Lern- und Steuerungszustand** — keinerlei Chatnachrichten, Formulierungen, Freitextantworten, Schülerlösungen, Bilder, Anhänge oder modellseitige Analyseprotokolle.

Die Lernkommunikation verschwindet dadurch nicht: Sie liegt beim MCP Host und (bei Vision-Nutzung) beim gewählten Model Provider. Weil heute kein einzelner Dienst alle Anforderungen gleichzeitig erfüllt (gehostete Agent Runtime, geräteübergreifende Chats, direkte Nutzer-Provider-Abrechnung, Open Responses, Remote MCP, Skills, MCP Apps UI, Agent Plugins, Mobile, Sprache, Vision und Minderjährigeneignung), verfolgt SkillPilot bewusst zwei Betriebswege parallel.

Standards und ihre Rollen

Standard / Begriff	Verbindliche Rolle in SkillPilot
Open Responses (vs. proprietäre *OpenAI Responses API* oder nachgebildetes `/responses`)	Offene Spezifikation zwischen Agent Runtime (MCP Host) und Model Provider/Gateway; Konformität zählt nur per Acceptance-Test und Capability-Profil, niemals bloß wegen des Pfadnamens `/responses`.
Model Context Protocol (MCP)	Standardisiert die Verbindung zwischen MCP Host/Client und **Skill Pilot Core** als Remote-MCP-Server über **Streamable HTTP**.
MCP Apps	Optionale interaktive HTML-Ressourcen in Host-Sandboxen. Jedes Skill Pilot-Tool unterstützt Textausgabe (universeller Fallback), `structured Content` und optionale MCP-Apps-UI; keine didaktische Pflichtaktion darf ausschließlich per UI möglich sein.
Agent Plugins v1	Portables Installationsartefakt (`plugin.json`, `skills/.../SKILL.md`, `mcp.json`); MCP-Apps-UIs kommen zur Laufzeit vom MCP-Server, und OAuth-Client-Profile bleiben hostverwaltet außerhalb des Plugins.

Laufzeit- und Installationssicht

```
Endgeraete (Mobile / Desktop Browser / App: Text, Diktat, Kamera)
        | HTTPS
        v
AI Chat Application / MCP Host
  +-- Chat UI + Server-side Agent Runtime (Loop, Context, Attachments, MCP Client)
        |                                        |
        | Open-Responses-API / Gateway           | Remote MCP (Streamable HTTP + OAuth)
        v                                        v
Model Provider (Nutzerkonto;            SkillPilot Core
  Vision-Bilder landen HIER,              +-- MCP Server (Tools, Resources, MCP Apps UI)
  nicht im SkillPilot Core)               +-- Didactic Services (Skill-Graph, SRS, Exams)
                                          +-- Learning State DB (nur strukturierter Zustand)
```

5.2 Zustandsgrenzen, Betriebswege, Medien und Abrechnung

Bereich	Gespeicherte Daten	Verantwortlich
MCP Host	Benutzerkonto, Chatverläufe, Nachrichten, Anhänge, Sitzungen, ggf. BYOK-Credentials	Hostbetreiber (im eigenen Pilot: enpasos)
Model Provider	Übermittelte Prompts und Vision-Bilddaten gemäß Anbieterbedingungen	Modellanbieter (direkter Vertrag mit Lernendem / Eltern)
SkillPilot Core	Strukturierter Lern- und Steuerungszustand (Zuordnung, Curriculum/Personalisierung, Fokus, `activeGoal`, binäre Mastery, SRS-Planung, binäres Verified-Recall-Ergebnis)	enpasos

- **Keine hostübergreifende Chatportabilität:** Der Chatverlauf ist hostgebunden, der Lernzustand SkillPilot-gebunden (`learner`, `activeGoal`, `masteryValue`, `dueItems`, `lastRecallOutcome`; `nextRecommendedAction` wird abgeleitet statt frei persistiert). Aktive Coach-Mutationen sind frei von Freitextfeldern (historische `lastFeedback`-Altlasten werden im aktiven Vertrag ausgeschlossen; siehe 6.2).
- **Sichtbare Kennungen:** Eine zufällige, opake `learningSessionId` ist im Modellkontext ausdrücklich erlaubt; permanente Lernenden-, Konto- oder Datenbank-IDs bleiben strikt verborgen.
- **Zwei dauerhafte Betriebswege: Betriebsweg A** nutzt fremdbetriebene Plattform-Hosts (ChatGPT, Claude) mit minimalem Eigenbetrieb und direktem Consumer-Abo; **Betriebsweg B** nutzt einen portablen Open-Source-MCP-Host (Pilot: LibreChat per Docker Compose; Zweitkandidat: Archestra) mit BYOK, direkter Providerabrechnung und kontrollierbarer Datenhaltung (inkl. dokumentierter Backup-, Upload-Lösch-, Key-Rotations- und Zugriffsregeln).
- **Sprache, flüchtige Bildanalyse und Abrechnung:** Diktat wird im Browser/Host zu Text gewandelt; fotografierte Lösungen gehen vom Host an das Vision-Modell des gewählten Providers und **niemals** an SkillPilot Core (Core erhält nur den strukturierten Schreibaufruf wie `setMastery` mit `1.0`). Providerkonto und Modell-Credentials gehören stets Lernenden bzw. Erziehungsberechtigten — ein zentraler SkillPilot-Betreiber-Key mit Weiterberechnung ist ausgeschlossen.

5.3 Stufenplan und verbindliche Host-Abnahmetests

1. **Phase 1 — Schnittstellen stabilisieren:** Hostunabhängiger Remote-MCP-Vertrag (Streamable HTTP, OAuth 2.0, stabile Toolnamen, versionierte Schemas, Idempotenz, Text-Fallback, `structuredContent`, optionale MCP Apps) plus kanonisches Agent-Plugin-Paket ohne Upload-Tools für Bilder/Lösungen.
2. **Phase 2 & 3 — Selbst betriebener Pilot & Standardslabor:** Kleiner geschlossener LibreChat-Pilot (5–10 Tester, Smartphone/Desktop, Diktat, Vision, BYOK) parallel zur praktischen Erprobung von Archestra (Open-Responses-Tests, MCP Apps).
3. **Phase 4 & 5 — Hosted Channels & künftiger Managed Host:** Claude (volljährige Tester) und ChatGPT (Verzeichnis-Release) auf demselben SkillPilot-Lernzustand; späterer Wechsel auf einen fremdbetriebenen portablen Host erst bei Nachweis aller 12 Kriterien (inkl. Open Responses, BYOK, MCP Apps, Agent Plugins, Datenexport/Löschung und Jugendschutz).

Pflicht-Abnahmetests für jeden Host und jedes Modell: Geräteübergreifende Sitzungsfortsetzung ohne Race-Conditions; 60 s deutsches Fachdiktat mit Tool-Call; Smartphone-Kamerafoto mit dokumentiertem Vision-Providerpfad; **MCP-Negativtest** (weder Bilddaten, OCR, Lösungstext, Bild-URL noch Analyseprotokoll in MCP-Argumenten, Logs, Traces oder DB); **Credential-Negativtest**; OAuth-Refresh und opake `learningSessionId` ohne interne ID-Lecks; Schreib-Idempotenz; Konto-/Chat-Löschtest; vollständiger Ablauf ohne MCP Apps sowie Sandbox-Prüfung mit MCP Apps; direkte Nutzerabrechnung; offizielle Open-Responses-Tests.

5.4 Teil B — Datierter Marktanhang (Stand 19. August 2026)

Kriterium	ChatGPT	Claude	LibreChat	Archestra	Managed Ziel
Runtime-Betrieb	OpenAI	Anthropic	selbst/fremd	selbst/fremd	fremd
SkillPilot mobil	im Verzeichnis zu prüfen	ja (iOS/ Android)	mobiles Web	zu testen	gefordert
Diktat & Kamera	Diktat/Bild ja; Live-Voice begrenzter	ja	ja (Vision direkt an Provider)	Anhänge dok.; mobil zu testen	gefordert
Abrechnung	Chat GPT-Abo	Claude-Abo	BYOK direkt	Provider-Keys	BYOK direkt
Open Responses	intern proprietär	nein	teilweise	`/responses`-Router zu prüfen	gefordert
MCP + Apps	Remote MCP; Custom Dev web-only	Remote MCP + Apps mobil	Remote MCP + PKCE; Apps Draft	Remote MCP + Apps	gefordert + Text-Fallback
Altersgrenze	ab 13 (13–17 mit Zustimmung)	Consumer ab 18 (API separat)	Betreiberregel	Betreiberregel	zielgruppengerecht
Rolle	strategischer Kanal	Referenz (ab 18)	**unmittelbarer Pilot**	**2. Kandidat**	Langfristziel

Open WebUI wird mangels nativer MCP-Apps-Kernunterstützung derzeit nicht gegenüber LibreChat oder Archestra bevorzugt. Verwandte Detailverträge: 6, 7, 8, 9, 10 sowie Open AI-MCP-Clientbindung.

6 Kommunikationsvertrag zwischen ChatClient und SkillPilot-Backend

Status und Geltungsbereich

Kanonische, providerneutrale Kommunikationsnorm für alle SkillPilot-Lerncoaches (OpenAI MCP App, Claude Connector V1, Gemini Connector V1, künftige Provideradapter und neue Coach-Werkzeuge). Sicherheits-, Session-, Didaktik- und Deploymentdokumente konkretisieren diese Grenze, dürfen sie aber nicht abweichend neu definieren. Eine erfolgreich getestete Provider-Lane (wie Geminis kontrollierter Web-Test im Gemini-Integrationsrunbook) belegt keine Abnahme anderer Hosts, Konten oder Oberflächen.

6.1 Kernsatz, Verantwortungsmatrix und minimaler Turn-Vertrag

Ein Satz legt die Grenze fest

Der ChatClient versteht und formuliert; das Backend weiß, entscheidet, validiert, speichert und liefert den nächsten technischen Schritt.

Ein Sprachmodell ist stark beim Verstehen freier Sprache, beim Erkennen fachlicher Muster, beim dialogischen Erklären und beim semantischen Vergleich gleichwertiger Lösungswege; das Backend ist stark bei Identität, aktuellem Zustand, exakten Mengen und Reihenfolgen, Berechtigungen, Transaktionen, Nebenläufigkeit, Persistenz und reproduzierbaren Entscheidungen.

Beteiligter	Besitzt	Besitzt ausdrücklich nicht
Lernende Person	Lernabsicht, Antworten, freie Formulierungen, ausdrückliche Zustimmung	technische IDs, Zustandsversionen, Retry-Schlüssel, Toolreihenfolgen
ChatClient / Modell	Sprachverstehen, eindeutige Intent-Zuordnung zu frischen Optionen, didaktischen Dialog, fachlich-semantischen Vergleich, Formulierung	autoritativen Lernzustand, Auswahlmengen, Workflowfortschritt, Persistenzerfolg, Berechtigung
Provideradapter	kleine use-case-orientierte Werkzeugoberfläche, sichere Projektion, lokalisierte Darstellung, Transportbindung	eigene Fachdatenbank, rohe Domain-DTOs, promptbasierte Sicherheitsgarantien
SkillPilot-Backend	Identität, Session, Curriculum, Personalisierung, Fokus, aktives Ziel, Frontier, Mastery, Recall, Prüfungsfreigabe, IDs/Capabilities, Mengen, Reihenfolge, State, Concurrency, Idempotenz, Persistenz, Fortsetzung	freie Sprachinterpretation oder semantische Beurteilung sichtbarer Lernendenarbeit
App-Widget	direkte deterministische Interaktion innerhalb seines engen UI-Workflows	Ersatz für Backendzustand oder allgemeine Chat-Orchestrierung

Die Interaktionssprache (`communicationLocale` des jüngsten autoritativen Vollergebnisses) ist ebenfalls Backendzustand und wird nie aus Toolnamen, Hostlocale oder Curriculumsprache erraten.

Der minimale Turn-Vertrag

```
neue Lernendennachricht -> genau ein frischer Vollzustand -> Entscheidung:
  +-- sprechen / fragen / gebundene UI-Aktion --> sichtbare Antwort
  +-- hoechstens eine eindeutige Mutation --> autoritativer Nachfolgerzustand
        +-- fortbestehende eindeutige Absicht erneut anwenden
        +-- sprechen / gebundene UI-Aktion --> sichtbare Antwort
```

1. Genau **ein** frischer Kontextabruf nach einer neuen Lernendennachricht; Polling im selben Assistant-Turn ist verboten.
2. Der jüngste erfolgreiche Vollzustand ist die alleinige Autorität für Sprache, Zustand, Optionen und erlaubte Aktionen.
3. Jede erfolgreiche Mutation liefert ihren vollständigen Nachfolgerzustand, der den vorherigen Kontext für den Rest des Turns ersetzt (kein redundanter Context-Reload).
4. Schmale Renderer- oder Widget-Receipts sind kein Vollkontext.
5. Bei State-Konflikt wird genau einmal frisch geladen und neu bewertet; wiederholte Konflikte oder harte Fehler stoppen den Ablauf transparent.

6.2 Werkzeuge und Eingaben: semantisch klein, frei von Chat-Freitexten

Werkzeuge bilden zusammenhängende Nutzerhandlungen ab (keine 1:1-Service-APIs): ableitbare technische Parameter fehlen im Modellinput, Item-Schleifen werden gebatcht, unterschiedliche Risiken/Bestätigungen bleiben getrennte Tools, und Reads vs. Writes sind klar annotiert.

Unverrückbare Grenze: keine Chat-Freitexte zum Core Der Coach beurteilt sichtbare Arbeit und formuliert Rückmeldungen selbstständig im Provider-Chat — weder wörtliche Antworten noch Paraphrasen oder Bewertungsbegründungen gehen an das Backend. Insbesondere sind Freitextfelder für Arbeits-, Ergebnis- oder Recall-Feedback in aktiven Coach-Mutationen verboten (unbekannte Zusatzfelder werden vor Domain-Aufruf, Replay-Hashbildung und Persistenz abgelehnt):

- **Normales Lernziel:** Strukturierte Abschlussentscheidung → Backend prüft aktives Ziel, Berechtigung und Regeln und bestätigt mit Ziel, Mastery-Wert und Folgezustand.
- **Verified Recall:** Ausschließlich die zum freigegebenen Batch gehörenden strukturierten `passed`-Booleans; Antworten und Begründungen bleiben im Chat.
- **Prüfung:** Autorisierte numerische Bewertung; Erläuterungen zu Lösungsweg, Punkten und Bestehen bleiben im Chat.

Didaktische Taktung: Abgeschlossene Aufgaben werden zuerst fachlich besprochen mit Angebot für Rückfragen oder Weitergehen; bei ausreichender Evidenz für ein gewöhnliches Lernziel oder eine bestandene Prüfung übermittelt der Coach die strukturierte Abschlussentscheidung sofort **vor** jeder Speicherbehauptung (Orientierung und Verified Recall behalten ihre gesonderte Zustimmung vor dem Speichern). Erst nach erkennbarer Zustimmung zur Fortsetzung beginnen Erklärung, Aufgabe und Lernbild des Backend-Nachfolgers. Lösungen offener Lernfragen werden nie als vorgefertigte Antwortvorschläge vorweggenommen, und Transportabbrüche/Timeouts werden sauber technisch aufgelöst statt als „Server nicht erreichbar“ erfunden.

Modellinputs enthalten ausschließlich semantisch zugeordnete Strukturwerte, fachlich-semantische Modellentscheidungen (`passed`) oder unverändert kopierte frische Optionspayloads/opake Capabilities (plus bei allgemeinen V1-Writes `expectedStateVersion` und frische `clientRequestId`).

6.3 Ergebnisse, opake Capabilities und Referenz Verified Recall

- **Ergebnisstruktur:** `structuredContent` enthält nur modellrelevante Fakten; `content` höchstens eine knappe Zusammenfassung; `_meta` ausschließlich widget-private Daten (kein Cross-Turn-Gedächtnis).
- **Ein einziger imperativer Kanal:** Normaler Vollkontext nutzt `(requiredAction, instruction)`; ein Spezialworkflow nutzt genau eine `continuation` (optional mit servergefülltem `toolCall`). Ein paralleles `presentationAction` oder eine implizite UI-Bindung an ein fachliches Write-Werkzeug ist nicht zulässig.
- **Opake Capabilities:** Kapseln Session-, Workflow- und Zustandsautorität (bei Vertraulichkeitsbedarf authentifiziert verschlüsselt), werden vom Modell unverändert durchgereicht und serverseitig auf Vollständigkeit, Stale State und Wiederverwendung geprüft.

Referenzablauf Verified Recall und terminaler Bild-Handoff

```
start_skillpilot_verified_recall(learningSessionId)
  -> kompletter serverseitiger Batch + batchCapability
get_skillpilot_verified_recall_answers(learningSessionId, batchCapability)
  -> komplette geordnete Sollantworten + gradingCapability
record_skillpilot_verified_recall_results(learningSessionId, gradingCapability,
assessments[{passed}])
  -> ein atomarer Receipt + genau eine continuation
```

Aktiviert ein terminaler Recall-Write ein gewöhnliches atomares Folgeziel mit autorisiertem Bild, lautet die einzige Fortsetzung:

```
{
  "continuation: {
    äction: "renderGoalVisualizationThenTeachActiveGoal",
    "consentRequired: false,
    "toolCall: {
      "name: "render_skillpilot_goal_visualization",
      ärguments: {
        "goalId: «server-filled successor goal>",
        ëxpectedStateVersion: 11
      }
    }
  }
}
```

Der ChatClient kopiert Name, `goalId` und `expectedStateVersion` unverändert und ergänzt ausschließlich die bereits aktuelle unveränderte `learningSessionId` gemäß dem globalen Session-Gate. Das Receipt spiegelt diese Session-Capability aus Datenschutzgründen nicht. Der Client ruft den servergefüllten `continuation.toolCall` für `render_skillpilot_goal_visualization` genau einmal unmittelbar auf und beginnt danach im selben Antwortturn das bereits aktive Folgeziel. Er rekonstruiert die bildspezifischen Argumente nicht aus `goalVisualization`, `nextAllowedTools` oder Top-Level-Metadaten und lädt keinen neuen Context. Scheitert das Rendering oder lässt der Host die Komponente weg, gibt es keinen Retry; der vollständige Textunterricht beginnt trotzdem. Der dedizierte Renderer bleibt das einzige an die Bild-UI gebundene Werkzeug; das Recall-Write selbst erhält keine UI-Bindung.

6.4 Auswahl, Fehler, Datenschutz, Anti-Patterns und 11-Punkte-Abnahme

- **Freie Auswahl & Fehler:** Mehrere sinnvolle Wege werden als kleine frische Optionsmenge veröffentlicht (nur bei Eindeutigkeit unverändert kopiert, bei echter Mehrdeutigkeit nachgefragt).

Fehler liefern stabile Codes ohne Teilmutation und lokalisierte Recovery-Hinweise; Sessionfehler enden ausschließlich mit der WebGUI-Startanweisung, während UI-Renderingfehler den Textpfad niemals blockieren.

- **Datenschutz & Anti-Patterns:** Permanente SkillPilot-ID, OAuth-Tokens, Secrets, interne DTOs/Pfade und ungeprüfte Lösungen erscheinen nie in Modellergebnissen. Verboten sind Item-Schleifen, optionale Modellparameter für Backendwissen, konkurrierende Imperativkanäle, redundante Context-Reads nach Mutationen, reine Prompt-Sicherheitsregeln, modellkonstruierte IDs/URLs, unbelegte Save-Behauptungen, Wortlaut-NLP im Backend sowie jeglicher Chat-Freitext in Toolinputs oder Persistenz.
- **Verbindliche Regelorte & 11-Punkte-Gate:** Jede Regel hat genau einen primären Ort (dieser Vertrag, 9, 7, 8, Einheitlicher Lernplanstatus, `SKILL.md`, Toolschema, Backendguard und `docs/deploy/`). Jede Schnittstellenänderung muss alle **11 Abnahmefragen** erfüllen: (1) eine Nutzerhandlung je Tool, (2) keine ableitbaren Parameter, (3) keine Modellschleifen, (4) genau eine Fortsetzung, (5) Backend-Ablehnung ungültiger/staler Aufrufe, (6) saubere Retry-Unterscheidung, (7) Receipt-belegte Aussagen, (8) Sprach-/Semantikarbeit beim Chat Client, (9) Contracttests inkl. Adversarial-Inputs, (10) realer Provider-Trace ohne Polls, (11) vollständiger Ausschluss von Chat-Freitexten aus Inputs, Echos und Persistenz.

7 SkillPilot-Lerncoach: Produkt- und Providerarchitektur

Status und Zusammenspiel der Verträge

Dauerhafte Produkt- und Systemtopologie des mehrsprachigen SkillPilot-Coaches. Arbeitsteilung und Turn-Vertrag stehen in 6, OAuth und 24h-Lernsession in 8, sichtbares Verhalten in 9, Versionierung inklusive der Entscheidung vom 19. Oktober 2026 für das dauerhafte neutrale Plugin `skillpilot-coach` (`https://mcp-openai.skillpilot.com/mcp`) in 10 und Betriebsabläufe unter `docs/deploy/`.

7.1 Produktentscheidung und mehrsprachige Plugin-Struktur

SkillPilot stellt den Coach als providergehostete Konversation mit authentisierten SkillPilot-Werkzeugen bereit:

- Modell, freier Chat und optionale App-Oberflächen laufen beim Provider über den bestehenden Consumerzugang der lernenden Person; SkillPilot ruft auf diesem Pfad keine eigene kostenpflichtige Modell-API auf und verkauft keine Modellnutzung weiter.
- Curriculum, Personalisierung, Fokus, Frontier, aktives Lernziel, Mastery, Recall, Prüfung und Persistenz verbleiben vollständig im SkillPilot-Backend.

Das mehrsprachige OpenAI-Plugin (`skillpilot-coach` im aktuellen neutralen Zielstand gemäß 10.1; historisch `skillpilot-coach-v1` in der unverändert bewahrten Git-Beta `1.1.2`) bündelt:

- einen neutral englischen Control-Plane-Skill,
- einen direkt gebundenen MCP-Server auf stabilem öffentlichem Origin (`https://mcp-openai.skillpilot.com/mcp`),
- einen sprachneutralen Werkzeugkatalog,
- zwei unabhängig gebundene, hashadressierte MCP-Apps-Ressourcen (Lernzielbild und normales Karteikartenlernen),
- getrennte reale Abnahmefälle je freigegebener Interaktionssprache und Oberfläche (Deutsch, Englisch und künftige Sprachen nutzen dasselbe Plugin über die in der Lernsession gebundene `communicationLocale`).

7.2 Systemtopologie und Schichtenverantwortung

```
SkillPilot-WebGUI (Level 2 konfigurieren, "Lernen starten")
  |  frische learningSessionId + vorbereitete Nachricht
  v
ChatGPT: SkillPilot Coach
  |  neutraler Skill, neutraler Toolkatalog, optionale MCP-Apps-UIs
  |  HTTPS + OAuth Bearer + learningSessionId
  v
https://mcp-openai.skillpilot.com/mcp
  v
OpenAI-V1-Provideradapter im Spring-Boot-Prozess (/internal/openai/v1/mcp)
  |  AuthN/AuthZ, Rate Limit, Schema, sichere Projektion
  v
CoachToolFacade + CoachStateProjection
  v
LearnerService + Curriculum + Datenbank
```

Provideradapter, Fachruntime und Domain laufen im selben `skillpilot-server`-Prozess ohne separates Integrationsgateway.

Schicht	Verantwortung
First-Party-WebGUI	Permanente ID, Providerhinweis, Level-2-Konfiguration, bewusster Start und Cockpit-Fallback
Provider-Host	Modell, Chat, Skillaktivierung, Toolauswahl und optionale Sandbox-UI-Darstellung
Plugin-Paket	Installierbares Artefakt aus Manifest, Skill, MCP-Verbindung und Releasebindung
Coach-Skill	Wiederholbares Dialog- und Coachingverhalten ohne eigene Zustandsautorität
Provideradapter	OAuth/Scopes, Toolkatalog, Schemas, Rate Limits, sichere Projektion und UI-Metadaten
`CoachToolFacade` & `CoachStateProjection`	Kleine providerneutrale Fach-Use-Cases und explizite Allowlist für sichtbaren Zustand
Domainruntime	Curriculum, Zustand, erlaubte Übergänge, Transaktionen, Idempotenz und Persistenz

7.3 Web-first-Einstieg, sichere Projektion und UI-Oberfläche

- **Web-first Level 2:** Permanente SkillPilot-ID, Curriculum/View, Bundesland, Dauermodell, Stufe, Fächer und Kursprofile werden ausschließlich im WebGUI erfasst; der V1-Coach-Vertrag fragt oder mutiert sie nie im Chat, sondern verweist bei unvollständiger Konfiguration zurück. Fokus und aktives Atomziel (Level 3) bleiben über frische Backendoptionen änderbar.
- **OAuth vs. 24h-Lernsession:** OAuth authentisiert die App; jeder bestätigte WebGUI-Start erzeugt unabhängig davon eine absolut auf 24 Stunden begrenzte `learningSessionId` (ohne Verlängerung durch Tokenrefresh; siehe 8).
- **Sichere Zustandsprojektion:** Provider sehen niemals rohe `UnifiedLearnerStateResponse`- oder Domainobjekte: permanente IDs und Provenienz werden entfernt, Aufgaben von Lösungen/Rubriken getrennt (Freigabe erst im geschützten aktiven Evaluations-Use-Case), URLs/Mathematik normalisiert, Fakten in `structuredContent` und widget-private Daten ausschließlich in `_meta` gelegt.
- **Zwei dedizierte MCP-Apps-Ressourcen:** `render_skillpilot_goal_visualization` bindet die aktive hashadressierte Bildressource; `start_skillpilot_memory_practice` bindet die interaktive Karteikartenressource (`release/line.json` ist die einzige Quelle aktiver Hashes; bereits ausgelieferte historische Bild-URIs bleiben passiv byteidentisch lesbar). App-only Kartenreview und alle übrigen Tools bleiben UI-ungebunden, und der vollständige Textpfad bleibt jederzeit funktionsfähig.

7.4 Bewertungsgrenze, weitere Provider und 8 Freigabegates

Das Provider-Modell vergleicht sichtbare Lernendenarbeit semantisch gegen freigegebene Referenzen/Rubriken; das Backend entscheidet allein über Freigabeberechtigung, aktives Ziel/Versuch, Vollständigkeit, Stale-/Replay-Schutz und Zustandsübergang. Weitere Provider (wie Claude oder Gemini) erhalten eigene Transport-/Auth-Adapter, geprüfte Toolkataloge und eigene End-to-End-Abnahmen über derselben `CoachToolFacade` und `CoachStateProjection`.

Vor einer öffentlichen Freigabe müssen **alle 8 Gates** gemeinsam grün sein: (1) Contract-, Schema-, Security- und Domain-Tests, (2) exakter Quell-/Draft-Abgleich, (3) OAuth-, Session-,

Ablauf- und Recovery-Tests, (4) reale Golden Journeys mit echtem Provider-Modell, (5) Desktop-, Mobile-Web- und relevante native Oberflächen, (6) jede freigegebene Interaktionssprache, (7) Datenschutz-, Minderjährigen-, Support- und Submissionprüfung sowie (8) Telemetrie, Rollback und unveränderte Cockpit-Nutzbarkeit.

8 OpenAI-MCP-App: OAuth- und 24h-Lernsession-Bindung

Geltungsbereich und Stand

Verbindliche Identitäts- und Sitzungsbindung des mehrsprachigen OpenAI-MCP-Coaches (**SkillPilot Coach** / `skillpilot-coach` auf `https://mcp-openai.skillpilot.com/mcp` gemäß der Entscheidung vom 19. Oktober 2026 in 10.1; historisch `skillpilot-coach-v1` auf `https://mcp-coach-v1.skillpilot.com/mcp` in der unverändert archivierten Git-Beta `1.1.2`). Für Plugin-Identität, Contract-Major und Lebenszyklus gilt ergänzend 10.

8.1 Architekturentscheidung: Zwei getrennte Berechtigungen

SkillPilot trennt App-Authentisierung und Lernendenadressierung strikt in zwei unabhängige Nachweise:

1. **OAuth authentisiert und autorisiert die MCP-Integration:** Die gehostete Verzeichnis-App nutzt genau einen vorregistrierten vertraulichen OAuth-Client (`client_id` + langes zufälliges `client_secret` via `client_secret_basic` am Token-Endpunkt, Authorization Code + PKCE `S256`). Für den autorisierten nativen Desktop-Pfad existiert daneben ausschließlich das exakt gepinnte öffentliche CIMD-Profil (`https://chatgpt.com/oauth/codex/7yTcOAOqrumX/client.json`, Callback `http://127.0.0.1/callback/7yTcOAOqrumX` mit dynamischem Port und PKCE `S256`), niemals als Fallback einer fehlgeschlagenen vertraulichen Authentisierung.
2. **Eine temporäre 24h-Lernsession wählt Lernenden und Interaktionssprache:** Erst ein ausdrücklich bestätigtes **Lernen starten** im First-Party-WebGUI erzeugt eine neue opake `learningSessionId` (`PT24H`) und öffnet einen neuen Chat. Die Zuordnung zur permanenten SkillPilot-ID und zur `communicationLocale` bleibt ausschließlich im Backend.

Für jeden fachlichen MCP-Toolaufruf müssen beide Bedingungen gleichzeitig erfüllt sein:

```
gueltiges OAuth Access Token (exakte Resource + Read/Write-Scope)
AND
gueltige learningSessionId (aktiv + Restlaufzeit >= PT1H)
```

OAuth allein erzeugt weder eine Lernsession noch wählt es einen Lernenden aus; eine `learningSessionId` allein autorisiert keinen MCP-Aufruf.

Bindung	Transport	Serverseitige Bedeutung
ChatGPT/App -> SkillPilot	OAuth-Code-Flow (`client_secret_basic` bzw. gepinntes Desktop-CIMD + PKCE `S256`), danach `Authorization: Bearer`	Genau das freigegebene Clientprofil darf ein auf die aktive MCP-Resource ausgestelltes Access Token beziehen und einlösen.
Chat -> Session -> Lernender	`learningSessionId` als Pflichtargument jedes fachlichen MCP-Tools	Diese gültige, im First-Party-WebGUI gestartete Session adressiert genau einen Lernendenzustand samt `communicationLocale`.

Die permanente SkillPilot-ID verlässt das Backend im MCP-Fluss niemals und erscheint weder im Chat noch in Toolargumenten oder -antworten.

8.2 Verbindlicher Startablauf, Erneuerung und Diagnose-TTL

Jeder Klick auf **Lernen starten** im First-Party-WebGUI ist eine eigene atomare Startoperation:

1. Das WebGUI kennt die gewählte SkillPilot-ID, die `communicationLocale` und den konfigurierten Level-2-/Level-3-Kontext.
2. Es sendet genau einen Startrequest an das SkillPilot-Backend.
3. Das Backend wendet den typisierten Startkontext atomar auf den autoritativen Lernendenzustand an.
4. Es erzeugt in genau diesem Augenblick eine neue kryptografisch zufällige `learningSessionId`.
5. Es persistiert ausschließlich deren HMAC-Hash zusammen mit interner `learner_id`, `communication_locale`, `started_at` und absolutem `expires_at`.
6. Es liefert eine vorbereitete Startnachricht und die Ziel-URL zurück; das WebGUI öffnet den neuen Chat mit eingetragener Nachricht:

```
Use SkillPilot Coach and continue.
learningSessionId: sps_<zufaelliger opaker Wert>
```

- **Jeder Start ist unabhängig:** Jeder Startrequest erzeugt eine neue `learningSessionId` — auch für denselben Lernenden, bei noch laufender älterer Session, bei geteiltem Chat GPT-Konto oder bestehender OAuth-Verbindung. Parallele Sessions bleiben unabhängig gültig; Nutzung verlängert `expires_at` niemals.
- **Erneuerung ausschließlich über WebGUI:** Ohne aktuelle Startnachricht ruft das Modell kein SkillPilot-Tool auf, sondern nennt lokalisiert `https://skillpilot.com/` und den WebGUI-Startweg. Liefert ein Tool `SESSION_REQUIRED`, `SESSION_RENEWAL_REQUIRED` oder `SESSION_VERSION_UNAVAILABLE`, gibt der Coach `instruction` unverändert aus (bzw. den exakten lokalisierten Eintrag aus `instructions`) und ergänzt `startUrl` nur, falls noch nicht enthalten — ohne Fachantwort, ohne Wiederverwendung der alten Session und ohne OAuth-Neuverbindung.
- **Gegatete First-Party-Diagnose-TTL:** Ausschließlich `POST /api/ui/learners/{skillpilotId}/openai/v1/launch` akzeptiert optional `diagnosticSessionTtlSeconds`, sofern `SKILLPILOT_OPENAI_COACH_V1_DIAGNOSTIC_SESSION_TTL_ENABLED=true` aktiv ist und der ganzzahlige Wert in `3601..86400` Sekunden liegt (nie über `PT24H`). Die Verkürzung (z. B. `3660` s für den Übergang unter `PT1H` nach 1 Minute oder `5400` s für einen 90-Minuten-Soak) gilt isoliert für genau diese eine Session; jeder Request ohne Feld läuft automatisch wieder mit `PT24H`.

8.3 MCP-Validierungsreihenfolge, Aktionshorizont und Verbote

Das Modell übernimmt `learningSessionId` unverändert als einfaches Pflicht-Stringfeld in jeden fachlichen Toolaufruf (Längen- und Formatregeln bleiben serverseitig, um Halluzinationen zu vermeiden), während der Connector `Authorization: Bearer <token>` außerhalb des Modellkontexts überträgt. Das Backend prüft strikt in dieser **8-stufigen Reihenfolge**:

1. gültiges OAuth Access Token;
2. exakte Resource/Audience (`https://mcp-openai.skillpilot.com/mcp`) sowie erforderlicher Read- oder Write-Scope;
3. vorhandene, syntaktisch gültige `learningSessionId`;
4. HMAC-basierte Auflösung in `openai_de_learning_session`;
5. Prüfung auf Ablauf und Widerruf;

6. Auflösung des autoritativen Lernendenzustands und Verfügbarkeit der gepinnten Contract-, Workflow- und Curriculumrevision;
7. Restlaufzeit `expiresAt >= now + PT1H` für jede Lese-, Schreib- oder Replay-Antwort (exakt `PT1H` ist zulässig; darunter Abbruch mit `SESSION_RENEWAL_REQUIRED`);
8. bei Writes Prüfung auf exakten idempotenten Replay (gleicher Toolname, kanonisch identische Argumente, identische `clientRequestId`, verfügbare gepinnte Versionen und `completedStateVersion == aktuelle Learner-Revision`; bei inzwischen fortgeschrittener Revision endet der Aufruf mit `STATE_VERSION_CONFLICT`).

Ausdrücklich verbotene Fallbacks: Eine fehlende oder ungültige `learningSessionId` darf niemals ersetzt werden durch ein OAuth-Subject, die „zuletzt erzeugte“ oder irgendeine andere Session desselben Lernenden, einen Pending Launch, eine im Chat genannte SkillPilot-ID oder eine beim OAuth-Callback implizit angelegte Session. OAuth-Callbacks, Token-Refreshes und MCP-Reconnects erzeugen, verlängern oder reaktivieren niemals eine Lernsession. Damit bleiben auch gemeinsame Familien- oder Schul-ChatGPT-Konten strikt getrennt.

8.4 Persistenzschema, OAuth-/mTLS-Härtung und Fehlerverhalten

Die Tabelle `openai_de_learning_session` bildet die alleinige Persistenzgrenze der kurzlebigen Sitzungsbindung (frühere OAuth-Subject-Schlüssel wurden verworfen):

Spalte	Bedeutung
`token_hash`	HMAC-Hash der ausgegebenen `learningSessionId` (Klartext wird nie gespeichert)
`learner_id`	Serverinterne Fremdschlüsselreferenz auf den Lernendendatensatz
`communication_locale`	Beim Start fixierte Sprache für Backendnutzdaten und sichtbare Coachantworten
`started_at`	Zeitstempel des bestätigten Klicks auf **Lernen starten**
`expires_at`	Absolute Ablaufzeit (`started_at + PT24H`, bzw. `+ diagnosticSessionTtlSeconds` im gegateten Test)

OAuth-Clientsecrets liegen ausschließlich in der geschützten Provider- und Serverkonfiguration (nie in Repo, Browser, Prompts, Argumenten, Antworten oder Logs). Offene Dynamic Client Registration (DCR) und ungepinnte CIMD-Dokumente sind deaktiviert; die exakte Callback-Allowlist, PKCE `S256`, Resource/Audience und Scopes werden erzwungen. Zusätzlich prüft der dedizierte MCP-vHost das OpenAI-Clientzertifikat (`observe`-Modus während der Integration: fehlende Zertifikate passieren bis OAuth, ungültige werden abgewiesen; `enforce`-Modus für erzwungene Kette, `clientAuth`-EKU und Connector-SAN; siehe OpenAI-MCP-Clientbindung). mTLS ersetzt weder OAuth noch die Lernsession.

Fehlersituation	Verbindliches Verhalten
OAuth fehlt / ungültig	Normale OAuth-Neuautorisierung durch den Host; erzeugt keine Lernsession
`learningSessionId` fehlt / unbekannt / manipuliert / abgelaufen / widerrufen	`SESSION_REQUIRED`; Coach gibt nur unveränderte `instruction` (ggf. mit `startUrl`) aus; keine Fachantwort, neuer WebGUI-Start + neuer Chat
Restlaufzeit < `PT1H`	`SESSION_RENEWAL_REQUIRED` (exakt `PT1H` bleibt gültig); neuer Web GUI-Start + neuer Chat
Gepinnte Workflow-/ Curriculumrevision nicht verfügbar	`SESSION_VERSION_UNAVAILABLE`; fail-closed, neuer WebGUI-Start + neuer Chat
Write-Scope fehlt	Ablehnung vor jeder fachlichen Mutation
Startkontext nicht atomar anwendbar / ungültige Diagnose-TTL	Startrequest ablehnen; keine `learningSessionId` ausstellen

8.5 Sicherheitsinvarianten und automatisierte Abnahmekriterien

Die Implementierung wahrt **17 Sicherheitsinvarianten**: (1) serverseitige permanente Skill Pilot-ID, (2) zufällige opake Session-IDs nur als HMAC-Hash, (3) absolute `PT24H`-Laufzeit außer im gegateten `3601..86400`-Diagnosefall, (4) keine Laufzeitverlängerung durch Nutzung, (5) `PT1H`-Aktionshorizont für neue Operationen, (6) identischer `PT1H`- und Revisionsschutz für idempotente Write-Replays, (7) frische unabhängige Session je Start, (8) keine Lernendenwahl durch OAuth allein, (9) keine MCP-Autorisierung durch Session allein, (10) `learningSessionId`-Pflicht für alle Fachtools, (11) kein OAuth-Subject-Lookup/-Fallback, (12) unveränderte Skill-Graph-/Frontier-/Mastery-Semantik, (13) strikte Clientprofil-Trennung (`client_secret_basic` bzw. exakt gepinntes Desktop-CIMD ohne Fallback), (14) Secret-Freiheit aller Artefakte/ Logs, (15) feste sessiongebundene `communicationLocale`, (16) Pflicht zu erfolgreichem `get_skillpilot_context` zu Beginn jedes Learner-Turns und direkter Nutzung des Mutations-Nachfolgerzustands im selben Turn sowie (17) Verbleib von permanenter ID, Providerhinweis und Level-2-Konfiguration im First-Party-WebGUI.

Automatisierte Tests müssen **alle 20 Abnahmekriterien** belegen:

1. Ein Klick auf **Lernen starten** erzeugt genau einen Startrequest und öffnet den Chat.
2. Jeder Start erzeugt eine neue `learningSessionId`, auch für denselben Lernenden.
3. Parallele Sessions verschiedener Lernender laufen kollisionsfrei über dieselbe OAuth-Verbindung.
4. Die Startnachricht enthält genau eine `learningSessionId` und keine SkillPilot-ID.
5. Sämtliche fachlichen MCP-Toolschemas verlangen `learningSessionId`.
6. Gültiges OAuth-Token ohne `learningSessionId` wird abgewiesen.
7. Gültige `learningSessionId` ohne gültiges OAuth-Token wird abgewiesen.
8. Unbekannte, manipulierte, widerrufene oder abgelaufene Sessions scheitern ohne Fallback.
9. Erfolgreiche Aufrufe lösen den Lernenden ausschließlich über die explizite `learningSessionId` auf.
10. Wiederholte Toolnutzung verändert `expires_at` nicht.
11. Read- und Write-Scopes werden pro Tool strikt durchgesetzt.
12. Lernziel-, Frontier- und Mastery-Tests bleiben unverändert grün.
13. Deutsch und Englisch nutzen denselben V1-Toolvertrag gesteuert über `communicationLocale`.
14. Token-Requests mit falschem/fehlendem Secret, fremder `client_id`, falscher Callback-URL oder fremder Resource werden abgewiesen.

15. Authorization-Server-Metadaten veröffentlichen das aktive Produktionsprofil ohne offene DCR oder ungepinnte Public-Fallbacks.
16. Das Diagnose-Feld akzeptiert `3601` und `86400` bei aktivem Gate und lehnt `3600`, `86401`, Werte oberhalb `PT24H` sowie Anfragen bei deaktiviertem Gate ab.
17. Diagnose-Starts (`3660` / `5400`) wirken isoliert auf eine Session; der nächste normale Start hat wieder `PT24H`.
18. Jeder Learner-Turn beginnt mit erfolgreichem Kontextabruf (Sessionfehler blockiert jeden Fachtext), und Mutationen liefern den im selben Turn direkt maßgeblichen Nachfolgerzustand ohne Zusatzread.
19. Der öffentliche V1-Toolkatalog enthält keine sessionlosen Start-, Capability-, Curriculum- oder Personalisierungstools.
20. Write-Replays unterhalb `PT1H`, bei fehlender gepinnter Version oder nach fortgeschrittener kanonischer Learner-Revision werden trotz identischer `clientRequestId` abgewiesen.

9 Verhaltensintegration des MCP-Lerncoaches

Auftrag und Abgrenzung

Lebendes, normatives Leitdokument für das **zusammenhängende sichtbare Verhalten** und die End-to-End-Abnahme des mehrsprachigen MCP-Lerncoaches: Aus Backend, Skill-Graph, Lernzustand, MCP-Vertrag und Provider-Modell wieder einen verlässlichen, für Lernende natürlichen Coach zu bilden. OAuth, MCP-Schema und vorhandene Tools allein beweisen kein Endnutzerverhalten — maßgeblich sind vollständige Nutzerreise, tatsächlicher Tooltrace, kanonischer Backendzustand und sichtbare Antwort zusammen. Einzelfehler führen zu allgemeinen Verbesserungen statt curricularen Sonderregeln. Ergänzende Verträge: 6, 8, 7, OpenAI-MCP-Clientbindung und Deployment-Runbook.

9.1 Erfolgsbild und nicht verhandelbare Invarianten

Aus Sicht der lernenden Person soll der Einstieg ohne Kenntnis von OAuth, MCP, Toolargumenten oder Sessiontokens funktionieren: (1) im First-Party-WebGUI auf **Lernen starten** klicken, (2) in einem neuen Chat mit eingetragener Startnachricht landen, (3) den Lernwunsch natürlich formulieren, (4) direkt im im WebGUI konfigurierten Level-2-Kontext weiterlernen (der Chat fragt oder ändert Level-2-Werte wie Curriculum, Bundesland, Dauer-/Jahrgangsmodell, Stufe, Fächer oder Kursprofile nie), (5) nur bei echter fachlicher oder Level-3-Mehrdeutigkeit gefragt werden, (6) am richtigen Atomziel im richtigen Modus lernen und (7) nach Unterbrechungen verlässlich fortsetzen.

Unveränderliche Invarianten über alle Provider- und Modellwechsel hinweg:

- **Backend und Skill-Graph:** Der jüngste erfolgreich geladene Backendzustand ist einzige Autorität; keine erfundenen Curricula, Ziele, Optionen, Fortschrittswerte oder Mutationen. Mastery hängt an der global eindeutigen Ziel-UUID über alle Ansichten hinweg. `requires` (gerichtet; Mastery eines abhängigen Ziels impliziert niemals rückwirkend Mastery offener Voraussetzungen) regelt Voraussetzungen, `contains` die Komposition; Motivationsziele sind nur Voraussetzung, wenn der Skill-Graph dies explizit modelliert. Frontier und Empfehlungen werden erst nach bestätigtem Lernumfang und Fokus bestimmt.
- **Appbindung und Lernsession:** OAuth authentisiert die App; jedes bestätigte **Lernen starten** erzeugt unabhängig davon eine frische, absolut 24 Stunden gültige `learningSessionId` (ohne Widerruf älterer Sessions; keine Ableitung aus OAuth oder Altchats). Jede Operation und jeder exakte idempotente Write-Replay benötigt mindestens `PT1H` Restlaufzeit (exakt `PT1H` ist zulässig), verfügbare gepinnte Workflow-/Curriculumversionen und beim Replay unveränderte kanonische Learner-Revision.
- **Nutzerkommunikation:** Keine Tool-, API-, JSON- oder Feldnamen in normalen Antworten; keine permanenten SkillPilot-IDs, OAuth-Tokens oder Secrets im Chat; keine Erfolgsaussage vor bestätigtem Backend-Receipt; im Prüfungsmodus keinerlei Hinweise oder Rückfragen.

9.2 Allgemeiner 10-stufiger Entscheidungszyklus, Scope vs. Fokus und Fehlerbehandlung

Bei Einstieg, Wiederaufnahme, Unsicherheit oder möglicher Kompaktierung folgt der Coach einem zustandsgebundenen **10-Schritte-Zyklus**:

1. **Lernsession übernehmen:** ausschließlich aus der aktuellen SkillPilot-Startnachricht.

2. **Frischen Zustand laden:** zu Beginn jedes Learner-Turns im aktuellen Assistant-Turn Kontext laden (nie aus Gesprächserinnerung rekonstruieren; nach erfolgreicher Mutation ist deren vollständiger Nachfolgerzustand für den Rest desselben Turns autoritativ ohne redundanten Zusatzread).
3. **Bestätigten Kontext bilden:** strikte Trennung zwischen bestätigtem Zustand, frischen Backendoptionen und bloßer Nutzerabsicht.
4. **Zulässige Nutzerabsicht auswerten:** nur fachliche Arbeit und explizite Level-3-Änderungen (Fokus/aktives Ziel), niemals Level 2.
5. **Eindeutige nächste Option semantisch zuordnen:** nur gegen Optionen des jüngsten Zustands nach Bedeutung, nicht nach Wortgleichheit oder Listenposition.
6. **Genau eine erlaubte Mutation ausführen:** keine Vorwegnahme späterer Schritte, keine konstruierten IDs.
7. **Frischen Nachfolgerzustand übernehmen:** direkt aus dem Mutationsresultat desselben Assistant-Turns.
8. **Absicht erneut auf den neuen Zustand anwenden:** solange genau ein Level-3-/Fachschritt eindeutig ist, ohne unnötige Zwischenfrage fortfahren.
9. **Nur echte Restmehrdeutigkeit erfragen:** zusammengehörige offene Angaben gebündelt mit knapp genanntem bestätigtem Kontext erfragen.
10. **Erst danach fachlich arbeiten:** Frontier, Zielwahl, Coaching, Recall oder Prüfung erst bei bestätigtem Lernumfang und Fokus.

Scope vs. Fokus: Der Coach unterscheidet Einstiegskontext (Curriculum/Paket), Personalisierung (Bundesland/Gültigkeit), Lernumfang (relevante Zielteilmenge), Fokus (aktueller Lernkorridor für Frontier/Optionen) und aktives Atomziel. Eine Angabe wie „Mathematik LK“ setzt im WebGUI nur Fach und fachbezogenes Kursprofil (z. B. Mathematik LK neben Physik GK), bestimmt aber weder Stufe (Sek I vs. Sek II) noch G8/G9 oder Phase; der Coach darf keinen Fokus auf der Gesamtwurzel belassen und daraus unbestätigte Stufen anbieten. Bei einer Fokusweitung veröffentlicht das Backend nur den learner-facing Vorfahrenpfad mit noch offenen `target`-Zielen (nächstgelegener Vorfahr zuerst); eine automatische Weitungsempfehlung erfolgt nur nach tatsächlichem Abschluss des aktuellen Fokus (leere Frontier allein genügt nicht) und wird erst nach Lernendenzustimmung gesetzt.

Begrenzte Fehlerbehandlung: Bei Konflikt genau einmal frisch laden und Absicht neu prüfen; bei `SESSION_REQUIRED`, `SESSION_RENEWAL_REQUIRED` oder `SESSION_VERSION_UNAVAILABLE` ausschließlich unveränderte `instruction` (bzw. lokalisierten Eintrag aus `instructions`) und `start Url` (falls noch nicht enthalten) ohne Fachantwort und ohne OAuth-Neuverbindung ausgeben; bei Auth-, Schema-, Speicher- oder wiederholtem Konfliktfehler transparent stoppen ohne erfundenen Ersatzpfad.

9.3 Wirksame Laufzeitorte und verbindliche Verhaltensregeln

Jede Regel besitzt genau einen primären Ort; technische Zähl-, Reihenfolge-, Vollständigkeits- und Idempotenzregeln werden durch atomare Backendoperationen statt mehr Prompttext gesichert (eingefrorene Custom-GPT- und Visible-Session-Pakete bleiben reine historische Rollback-Baselines):

Regelart	Primärer Ort
Sprache vs. technische Orchestrierung	Kommunikationsvertrag (6)
Sichtbares didaktisches Coach-Verhalten	Dieses Kapitel (9)
Wiederholbares Modellverhalten	Ausgelieferte `SKILL.md` und `references/coaching-policy.md` (`ai/openai plugin/skillpilot-coach/`)
Werkzeugübergreifende Invariante	Kurze `SERVER_INSTRUCTIONS` in `OpenAiDeV1McpContractAdapter`
Auswahl und Bedingung eines Werkzeugs	Toolname, Beschreibung und Schema im V1-Adapter
Zustandsabhängige Einzelentscheidung	Genau ein frisches Toolergebnis aus `OpenAiDeCoachContext Projector` bzw. Workflowadapter
Fachliche & sicherheitsrelevante Garantie	Domainservice, Guard, Transaktion und automatisierter Test
Reales Zusammenspiel	Golden Journey, Tooltrace, Backendzustand und sichtbare Antwort

Policy-ID	Verbindliche Regel
`COACH-STATE-001`	Der frisch geladene Backendzustand ist die einzige Zustandsautorität.
`COACH-SESSION-001`	Die aktuelle Lernsession wird unverändert für jeden fachlichen Aufruf verwendet und nie aus OAuth oder älteren Chats abgeleitet.
`COACH-SESSION-002`	Operationen und gespeicherte Write-Replays benötigen mindestens `PT1H` Restlaufzeit (exakt `PT1H` gültig), verfügbare gepinnte Workflow-/Curriculumversionen und beim Replay unveränderte kanonische Learner-Revision ohne erneute Mutation.
`COACH-SESSION-003`	Sessionfehler ergeben ausschließlich die unveränderte Serverinstruktion und, falls noch nicht enthalten, die exakte `startUrl`; keine Fachantwort oder OAuth-Neuverbindung, Fortsetzung über WebGUI-Start und neuen Chat.
`COACH-BOOTSTRAP-001`	Ohne aktuelle vorbereitete Session nennt der Coach nur `https://skillpilot.com/` und den WebGUI-Startweg und ruft kein SkillPilot-Tool auf.
`COACH-INTENT-001`	Natürliche mehrteilige Absichten gelten unabhängig von Reihenfolge und Wortlaut fort.
`COACH-CONTEXT-001`	Vor offenen Fragen wird der bereits bestätigte fachliche Kontext knapp genannt.
`COACH-SCOPE-001`	Level-2-Lernumfang und Profile werden ausschließlich im WebGUI aufgelöst; der Coach verwendet sie nur frisch bestätigt.
`COACH-FOCUS-001`	Zieloptionen stammen ausschließlich aus dem bestätigten aktuellen Fokus; frühere Stufen dürfen nicht hineinlecken.
`COACH-MUTATION-001`	Pro frischem Zustand wird nur eine aktuell erlaubte Option mutiert; ihr vollständiger Nachfolgerzustand ist für den Rest desselben Assistant-Turns autoritativ.
`COACH-QUESTION-001`	Der Coach fragt nur echte Restmehrdeutigkeiten und fasst zusammengehörige offene Angaben möglichst zusammen.
`COACH-GOAL-001`	Unterricht findet an genau einem bestätigten atomischen Ziel statt.
`COACH-MASTERY-001`	Mastery folgt der global eindeutigen Lernziel-ID und wird nur nach ausreichender Evidenz gespeichert.
`COACH-RECALL-001`	Das Backend besitzt IDs, Kartenzahl, Reihenfolge, Vollständigkeit, Status, Idempotenz und Fortsetzung eines Recall-Batches; das Modell zeigt den gesamten Batch, wartet auf alle Antworten, lädt alle Sollantworten genau einmal, vergleicht fachlich-semantisch und speichert alle Bewertungen genau einmal atomar ohne Kartenschleifen.

Policy-ID	Verbindliche Regel
`COACH-EXAM-001`	Prüfung bedeutet wortgetreue Aufgabe, keine Hilfen oder Rückfragen und faire kriteriumsbezogene Bewertung gleichwertiger Wege.
`COACH-RESOURCE-001`	Fachliche Ressourcen stammen nur aus dem frischen Zustand: frisch autorisierte Zielvisualisierung genau einmal unmittelbar über ihren gebundenen Renderer, normale Kartenpraxis über ihre gebundene Kartenkomponente, Cockpit-Fallback nur in den dokumentierten Fällen.
`COACH-ERROR-001`	Fehlerbehandlung ist begrenzt, wahrheitsgemäß und erzeugt keinen erfundenen Ersatzablauf.
`COACH-PRIVACY-001`	Technische Identitäten, Geheimnisse und interne IDs bleiben außerhalb sichtbarer Coachantworten.

10 abstrakte Fehlerklassen für ursachenbezogene Fixes statt Einzelfall-Patches: (1) Session/ Handoff, (2) Zustandsprojektion, (3) Intentverständnis, (4) Scope-Auflösung, (5) Fokussetzung, (6) Toolverständlichkeit, (7) Orchestrierung, (8) Kontextverlust, (9) Didaktik/Modus und (10) Nutzerkommunikation.

9.4 Golden Journeys (`GJ-01` bis `GJ-11`)

- `GJ-01` – **Frischer Start:** Ein Klick auf **Lernen starten** im WebGUI erzeugt eine neue 24h-Lernsession, öffnet einen neuen Chat mit vorbereiteter Nachricht und lädt im ersten Assistant-Turn den Kontext vor jeder Fachantwort (kein Zweitklick, kein manuelles Kopieren, keine generische Lehrplanantwort).
- `GJ-02` – **Level-2-Änderungswunsch im Chat:** Bei Eingaben wie „Hessen, Mathe LK“ erfolgt keine chatseitige Curriculum-/Personalisierungsmutation und keine Rückfrage zur Level-2-Einrichtung; knapper Verweis auf den serverseitig gelieferten WebGUI-Weg und frischer Start nach Konfiguration.
- `GJ-02a` – **Mehrere fachbezogene Kursprofile:** Sind im WebGUI z. B. Mathematik LK und Physik GK gespeichert, liest der Coach beide fachbezogenen Profile unverändert aus dem frischen Kontext, ohne ein globales Profil anzunehmen oder die Stufe aus LK/GK abzuleiten.
- `GJ-03` – **Orientierung und Fokus:** Bei neu bestätigtem aktivem Ziel beginnt die erste inhaltliche Antwort mit dem exakten lokalisierten `activeGoal.title` (nicht der Beschreibung); Zieloptionen stammen ausschließlich aus dem bestätigten Fokusunterbaum ohne Hineinlecken früherer Stufen oder der Gesamtwurzel.
- `GJ-03a` – **Aktiver Motivationsdialog:** Erster Zug: exakter Zieltitel, 2–4 altersgerechte Möglichkeiten, positive Perspektiven und niedrigschwellige persönliche Interessenfrage ohne Richtig/Falsch. Zweiter Zug auf die gewählte Option (z. B. „Smartphone und KI“): konkrete Verknüpfung mit 1–2 Dingen zum Verstehen/Gestalten und persönliche Anschlussfrage bei weiterhin aktivem Orientierungsziel. Abschluss nach Antwort (oder ausdrücklichem Weiterlernwunsch): positive Zusammenfassung, Frage nach Rückfragen/Abschluss, Speicherung erst nach Zustimmung (technischer Abschlussmarker ohne Fachkompetenzbescheinigung; keine Wissensdiagnose oder Feynman-Prüfung).
- `GJ-04` – **Normaler Coachingzyklus:** Ein aktives Atomziel, Vorwissensdiagnose, kleine Hinweise, eigenständige Arbeit, Transfer und faire Rekonstruktion alternativer Wege. Vor der Rückmeldung entscheidet der Coach anhand unabhängiger Evidenz zwischen Aufgaben- und Zielabschluss (eine gelöste Aufgabe allein beweist nicht das ganze Ziel; bei ausreichender Zielevidenz sofortige Mastery-Speicherung vor Erfolgsmeldung, auch vor einer Pause; Zustimmung zum Weitergehen ist keine Speicherbedingung, aber Voraussetzung für nächste Aufgabe samt Bild). Unterbrechungen setzen denselben bestätigten Schritt fort.

- `GJ-05` – **Verified Recall:** Vorbereiteter Batch (geprüft mit 8, 5 und 10 Karten): genau ein Aufruf von `start_skillpilot_verified_recall` ohne modellseitige `goalId` oder `batchSize`; Anzeige exakt aller gelieferten Fragen in Originalreihenfolge; Abwarten aller Antworten; genau ein `get_skillpilot_verified_recall_answers`; semantische Bewertung äquivalenter Formulierungen, Gesamtrückmeldung und Abwarten der Zustimmung nach der Frage nach Rückfragen oder Abschluss; danach genau ein atomarer Write `record_skillpilot_verified_recall_results` mit je einer Bewertung pro Karte — ohne per-card Toolschleife, ohne manuelle Mastery und ohne weiteren Context-Abruf im selben Lernendenzug. Will die lernende Person weitergehen, setzt der Coach die eine bestätigte Backend-`continuation` unverändert um (bei Pause kein nächster Inhalt). Aktiviert der terminale Write ein Folgeziel mit freigegebenem Bild, lautet die Aktion `renderGoalVisualizationThenTeachActiveGoal`: Der Coach führt den vollständig servergefüllten `continuation.toolCall` für `render_skillpilot_goal_visualization` mit `goalId` und `expectedStateVersion` unverändert genau einmal nach Zustimmung zum Weitergehen aus, ergänzt nur die bereits aktuelle unveränderte `learningSessionId` gemäß Session-Gate und beginnt danach das aktive Ziel. Das Receipt spiegelt die Session-Capability nicht. Der Coach leitet die bildspezifischen Argumente nicht selbst aus dem Successor-Context ab, lädt keinen neuen Context und versucht es bei Renderer- oder Hostfehlern nicht erneut, sondern setzt den vollständigen Textunterricht fort. Verboten sind Teilmengen (wie 5 von 8 Fragen), eigene ID-/Reihenfolgewahl, einen parallelen `presentationAction` zu verwenden oder die Recall-Mutation selbst an die Bild-UI zu binden.
- `GJ-06` – **Prüfung:** Wortgetreue Aufgabe ohne Hilfen/Nachfragen, Schwelle und Bewertungsraster erst nach vollständiger Abgabe, geforderte Zeichnungen als echte Zeichnungen, kriteriumsbezogene Bewertung inklusive gleichwertiger Wege. Bestandener Versuch speichert Mastery sofort mit freigegebener Evaluation-Capability und Punktzahl (auch bei Pausenwunsch) vor der Erfolgsmeldung; nicht bestandene Versuche schreiben nichts und bleiben wiederholbar. Danach Teilpunkte, Gesamtergebnis, Feedback und Frage nach Rückfragen/Weitergehen; fehlende Pflichtabbildung pausiert dieselbe Prüfung ohne erfundene Bilddaten.
- `GJ-07` – **Langdialog und Rehydration:** Nach 10–20 Zügen, Nebenthemen, Reload oder Kontextdruck lädt die gültige Session denselben Backendzustand frisch, ohne alte Optionen zu mutieren.
- `GJ-08` – **Aktionshorizont, Ablauf und Erneuerung:** Unterhalb `PT1H` (exakt `PT1H` bleibt zulässig), nach Ablauf/Widerruf oder bei fehlender gepinnter Revision erscheint ausschließlich die unveränderte Serverinstruktion (ggf. mit `startUrl`) ohne Fachantwort, ID-Abfrage oder OAuth-Neuverbindung; WebGUI-Start öffnet frische Session und neuen Chat.
- `GJ-09` – **Parallele Lernende und Chats:** Verschiedene UI-Starts und SkillPilot-IDs bleiben strikt isoliert.
- `GJ-10` – **Andere Curricula:** Derselbe Entscheidungszyklus trägt über Schulformen, Länder und Fächer ohne curricularen Sondercode.
- `GJ-11` – **Gebundene UI-Aktionen und Cockpit-Grenze:** Jedes frisch autorisierte `(goalId, stateVersion)`-Paar triggert genau einmal unmittelbar den Zielbild-Renderer; normale Kartenpraxis startet die gebundene Kartenkomponente; Cockpit-Fallback erscheint nur bei echtem Komponentenfehler, fehlender Berechtigung, explizitem Nutzerwunsch oder Serveranweisung.

9.5 Testpyramide, Qualitätsziele, Testprotokoll und Definition of Done

Die **5-stufige Testpyramide** kombiniert (1) deterministische Vertragstests, (2) LLM-freie Zustands-/Workflowtests, (3) MCP-Protokolltests, (4) Model-in-the-Loop-Goldensuite mit mindestens 3 Paraphrasen je Kernabsicht und aufgezeichneter Toolspur sowie (5) Langdialog-, Reload- und Kompaktierungstests. Serververifizierbare Toolspuren und Vorher/Nachher-Zustände schlagen Modellselbstauskünfte.

- **Harte Gates:** 100 % korrekter Backendendzustand sowie Scope/Fokus in allen freigegebenen Golden Journeys, null unerlaubte Mutationen, null Session-/Lernendenvermischung, null Geheimnis-/Identitätslecks, null Regression der globalen Ziel-/Mastery-Semantik.
- **Qualitätsziele:** >= 3 semantisch verschiedene Formulierungen je Kernabsicht, >= 95 % erfolgreiche reale Dialogläufe je Szenario, <= 1 nachweislich unnötige Rückfrage, null wiederholte bereits beantwortete Fragen, null sichtbare API-Sprache, konstante Qualität über kurze und lange Dialoge.

Protokollfeld	Inhalt bei manueller Erprobung
Test-ID & Golden Journey	z. B. `GJ-02 / INTENT-003`
Datum, Umgebung & Versionen	Umgebung (lokal/Staging/Prod) + Git-SHA, Server-Build, Paket- und Vertragsversion
Lernfixture & Startart	Curriculum, pseudonymisierte SkillPilot-ID, Vorzustand, UI-Start / Reload / Ablauf
Wortgetreue Nutzereingaben	Exakte Eingaben in Originalreihenfolge
Erwartete vs. reale Spur	Erwartete Teilordnung + verbotene Calls vs. Telemetrie-/ Serverlog-Spur
Backendzustand & Sichtbarkeit	Vorher/Nachher-Zustand + fachlich-kommunikative Rubrikbewertung
Fehlerklasse, Policy & Fix	Betroffene Klasse/Policy-IDs, allgemeiner Fix (kein Einzelfallpatch) und Retest-Status

Ausdrückliche Nicht-Ziele: keine Verbiegung der globalen Lernziel-/Mastery-Semantik für einzelne Dialoge, keine Hessen-/Mathe-/LK-Sonderlogik, keine 1:1-Kopie alter HTTP-Routen, keine Steuerung über die öffentliche Listing-Beschreibung, kein Vertrauen auf Modellmemory und keine Behauptung der Endnutzerreife allein aus grünen Schematests.

Definition of Done (10 Kriterien): (1) harte Gates für alle Golden Journeys erfüllt, (2) geschlossene Traceability je Policy-ID, (3) mehrere Curricula und natürliche Mehrfachangaben ohne Sondercode grün, (4) Scope und Fokus vor jeder Zielausgabe korrekt, (5) Unterricht, Mastery, Recall und Prüfung fachlich abgenommen, (6) Langdialog/Reload/Kompaktierung/Parallelchats verlust- und vermischungsfrei, (7) Ein-Klick-Einstieg ohne manuelle Technikwerte, (8) sichtbare Kommunikation erfüllt Qualitätsziele, (9) CI, reale Providertests, Telemetrie und Rollback grün sowie (10) früherer Custom-GPT-Coach nur noch historische Referenz.

10 SkillPilot: Versionierungs- und Lebenszyklusplan für das OpenAI-Plugin

Geltungsbereich und Release-Einordnung

Verbindliche interne Architekturgrundlage für Versionierung, dedizierte hashgebundene MCP-Apps-Ressourcen, gebündelte Skills und den Lebenszyklus des OpenAI-Plugins. Die veröffentlichte Git-Beta `skillpilot-coach-v1 1.1.2` und ihre Installationsbytes sowie die verworfene `1.0.0`-Reviewhistorie (Review-Freeze-Historie) bleiben unverändert archiviert. Der neutrale lokale Kandidat `1.1.3` (einschließlich des lokalen Handshake-Fixes `9c09821e96ab…` neben den unverändert bewahrten `12762009…`- und produktiv ausgerollten `08030…`-Bytes; Voice-Bildfolge weiterhin ohne Hostabnahme) ist Vorbereitung vor der ersten Verzeichniseinreichung — weder Deployment noch Portal-Veröffentlichung oder Clientabnahme.

10.1 Entscheidung vom 19. Oktober 2026: ein dauerhaftes neutrales Plugin

Der Product Owner hat am **19. Oktober 2026** ein dauerhaftes Plugin **SkillPilot Coach** mit dem technischen Namen **`skillpilot-coach`** und einem neutralen öffentlichen MCP-Origin vor der neuen Verzeichniseinreichung festgelegt. Diese Entscheidung ersetzt sowohl die erste Empfehlung, `skillpilot-coach-v1` und dessen Host beizubehalten, als auch das frühere Modell einer eigenen Plugin-Identität pro Paket-Major.

Grenze	Dauerhafte Festlegung
Technischer Paketname	`skillpilot-coach`
Öffentlicher Anzeigename	`SkillPilot Coach` (<= 30 Zeichen)
Öffentlicher MCP-Endpoint & OAuth Resource/Audience	`https://mcp-openai.skillpilot.com/mcp`
Widget-Origin (`_meta.ui.domain`)	`https://mcp-openai.skillpilot.com`
Protected Resource Metadata	`https://mcp-openai.skillpilot.com/.well-known/oauth-protected-resource/mcp`
Bestehender OAuth-Issuer	`https://skillpilot.com/api/openai/v1`
Aktueller interner Adapter	`/internal/openai/v1/mcp`, `contractMajor = 1`

Paket-SemVer, öffentliche Identität und fachlicher Toolvertrag sind entkoppelt: Künftige Paketversionen `2.x` können denselben Toolvertrag nutzen, und interne Workflowgenerationen erfordern keinen neuen öffentlichen Namen oder Origin, weil OpenAI Origin-Wechsel im Updateablauf veröffentlichter Plugins nicht unterstützt.

- **Begrenzter technischer Umbau & Desktop-CIMD:** Interner V1-Adapter, Scopes und Session-Gates bleiben erhalten (`v1` im OAuth-Issuer bezeichnet die bestehende Autorisierungsinstanz, keine Paket-Major-Vorgabe). MCP-URL, Resource/Audience, Widget-Origin, Metadata-URL, `/.well-known/openai-apps-challenge` und Edge-Host wechseln gemeinsam auf `https://mcp-openai.skillpilot.com` — mit genau einer aktiven Resource ohne Dual-Audience oder Alt/Neu-Dispatcher. Native Desktop-CIMD nutzt exakt den zur neuen MCP-URL gehörenden öffentlichen Client `https://chatgpt.com/oauth/codex/7yTcOAOqrumX/client.json` und Callback `http://127.0.0.1/callback/7yTcOAOqrumX` (dynamischer Port, PKCE `S256`), strikt getrennt vom vertraulichen Clientprofil ohne Fallback.

- **Cutover & Rollback (`AuthenticatedClientPolicy`):** Alte Chats sind ausdrücklich ausgeschlossen (keine Chat-Fortführung, Sessionmigration oder parallele Alt-MCP-Runtime erforderlich; dauerhafte Lernstände bleiben erhalten). Die OAuth-Resource ist Teil des Clientprofil-Fingerprints: Nach erfolgreicher Startvalidierung aktiviert `AuthenticatedClientPolicy` den neuen Fingerprint und sperrt den vorherigen ohne Grant-Umschreibung (alte und neue Instanzen dürfen beim Cutover nicht gleichzeitig dieselbe DB mit verschiedenen Profilen bedienen). Ein Rollback auf einen gesperrten Fingerprint wird verweigert und erfordert eine neue explizite `policyRevision`, erneute Autorisierung sowie Kompatibilität mit allen freigegebenen Live-Tooldefinitionen.
- **Spätere Tool-Evolution in einer Identität:** Paket, Skills, Server und einzelne Tooldefinitionen aktualisieren sich beim Provider asynchron; während der Prüfung einer neuen Definition kann die vorherige weiter aktiv sein. Inkompatible Toolwechsel nutzen deshalb eindeutig benannte Ersatzwerkzeuge oder kompatible Übergangsadapter, bis die freigegebenen Live-Definitionen das Entfernen erlauben (`clientInfo.version`, User-Agent und Prompttext steuern niemals Autorisierung oder Vertragsrouting).

10.2 Unabhängige Versionsebenen und SemVer-Klassifikation

Keine einzelne Versionsnummer darf für mehrere Ebenen missbraucht werden:

Versionsebene	Beispiel	Bedeutung & Quelle der Wahrheit
Plugin-Identität	`skillpilot-coach`	Dauerhafte öffentliche Verzeichnisidentität und stabiler MCP-Origin
Plugin-Paketversion	`1.1.3`	SemVer des Pakets aus Manifest, Skills und geprüften MCP-/UI-Metadaten
MCP-Tool-Contract-Major	`1`	Semantik und JSON-Schemas des internen Vertragsadapters (`v1`)
MCP-Protokollversion	datumsbasiert	Transport-/Handshake-Version zwischen Host und MCP-Server
Server-Build	Git-SHA / Buildnr.	Konkret deploytes Backend-Artefakt `skillpilot-server`
UI-Artefaktversion	`sha256-...`	Unveränderliche content-adressierte MCP-Apps-HTML/JS/CSS-Auslieferung
Zustands-Schemaversion	`3`	Internes Persistenzformat (über interne Datenmigrationen entkoppelt)
Zustandsversion (`state Version`)	`142`	Monoton steigende Revision des kanonischen Lernendenzustands über alle Sessions und Web GUI hinweg
Workflow-Version	`coach@2.3`	Gepinnte pädagogische Ablauf- und Entscheidungslogik einer Session
Curriculum-Revision	`curricula-sha256@...`	Gepinnte Lernziel- und Skill-Graph-Revision

Draft vs. Published & SemVer-Schritte:

- **Unveröffentlichte Arbeitsstände (`DRAFT`):** `contracts/drafts/openai/<plugin>/<version>-SNAPSHOT/` ist vor `Submit for Review` über `prepare` reproduzierbar fortschreibbar (Manifest trägt `<version>` ohne `-SNAPSHOT`); während eines laufenden Portal-Reviews schützt eine Review-Sperre Paket und Runtime. `contracts/published/openai/<plugin>/<version>/` und `release-index.json` entstehen ausschließlich nach bestätigtem Portal-Publish über den expliziten Schritt `record-published` und sind danach unveränderlich.

Deterministische USTAR-Installationsarchive (`plugin-install-bundle`) und Verzeichnis-ZIPs hängen weder von lokaler `tar`-Version noch von `umask`, Dateirechten oder Symlinks ab und enthalten nie den Server.

- **Reiner `serverBuild`-Wechsel (kein Paket-Bump):** Reine Bugfixes, Performance-, DB- oder interne Sicherheitskorrekturen bei identischem sichtbarem Auth-, Tool-, Schema-, Skill- und UI-Vertrag (alle Contract-Tests bleiben grün).
- **PATCH (`x.y.z -> x.y.(z+1)`):** Präzisierung von Toolbeschreibungen, `SKILL.md` oder Annotationen auf bereits bestehendes Verhalten, kompatible UI-Korrektur unter neuer Hash-URI oder Listing-/Release-Notes-Korrektur.
- **MINOR (`x.y.0 -> x.(y+1).0`):** Rückwärtskompatible Erweiterungen (neues Tool, optionales Eingabefeld, Erweiterung in `extensions`, neuer Skill, neue UI-Ressource, additiver OAuth-Scope); der Live-Server wird **vor** dem Portal-Scan als Superset für Alt- und Neuvertrag deployt.
- **Inkompatible Änderungen:** Entfernen/Umbenennen von Tools oder Feldern, neue Pflichtfelder, Typ-/Semantikbrüche oder schreibende Umwandlung eines Read-only-Tools dürfen niemals ungeschützt live gehen; sie erfolgen über eindeutig benannte Ersatztools/Superset-Adapter oder eine explizite neue Vertragsgeneration. Skills liegen als versionsgeprüfter Dateibaum im Paket (`Skill-Contract-Lint` erzwingt existierende Toolnamen derselben Vertragslinie und reproduzierbare Hashes).

10.3 Öffentliche Vertragsadapter, Envelopes und dedizierte MCP-Apps-UIs

Öffentliche MCP-Schemas exportieren niemals rohe Domain- oder Persistenzklassen. Zustandsbezogene Antworten und schreibende Aufrufe verwenden explizite Versions- und Idempotenzfelder (`additionalProperties: false` im Kernschema, offene kompatible Erweiterungen unter `extensions`, strukturiertes Resultat plus knappe Textdarstellung, niemals interne Benutzer-IDs):

```
{
  "contractMajor: 1,
  "learningSessionId: ßps_...",
  ßtateVersion: 42,
  ßtateSchemaVersion: 3,
  "workflowVersion: "coach@2.3",
  "curriculumRevision: "curricula-sha256@...",
  "currentStep: { "goalId: "..." },
  ällowedActions: [ßubmit_answer"],
  ëxtensions: {}
}
```

Schreibaufrufe übergeben `learningSessionId`, `expectedStateVersion` (Optimistic Locking gegen konkurrierende Transporte) und `clientRequestId` (idempotente Retries innerhalb des `PT1H`- und Revisionsfensters aus 8); gepinnte `workflowVersion`- und `curriculumRevision`-Stände halten laufende Sessions konsistent oder stoppen fail-closed mit `SESSION_VERSION_UNAVAILABLE`.

Zwei dedizierte V1-MCP-Apps-Ressourcen (`text/html;profile=mcp-app`):

1. `render_skillpilot_goal_visualization` (read-only) bindet über `ui.resourceUri` und `openai/outputTemplate` ausschließlich die aktuelle bild-only Ressource und wird je frisch freigegebenem `(goalId, stateVersion)`-Paar genau einmal aufgerufen (`expectedStateVersion = stateVersion`), während der vollständige Textpfad erhalten bleibt.
2. `start_skillpilot_memory_practice` (read-only) bindet ausschließlich die eigene interaktive Karteikarten-Ressource.

3. Das app-only Bewertungstool `review_skillpilot_memory_practice_card` sowie alle übrigen Fachtools bleiben UI-ungebunden (keine verschachtelten oder leeren Komponenten; autoritativer Zustand liegt ausschließlich im Backend; bei beschädigtem Hostzustand sicherer sichtbarer Fehlerzustand statt lokalem Weiterlernen).
4. **Passive Byte-Aufbewahrung ab erster Realauslieferung:** Jede content-adressierte URI (z. B. `ui://skillpilot/coach/1/sha256-<hash>/index.html`), die je einem realen Client angeboten wurde, bleibt mit byteidentischem Inhalt passiv lesbar, während im Tool-Descriptor ausschließlich die aktuelle aktive URI aus `release/line.json` gebunden ist.

10.4 Kanonisches `contract-line/v1`-Schema, CI-Gates, Observability und Linienhistorie

Support-Lifecycle, Publikationsstatus und Startpolicy bilden drei unabhängige, serverautoritative Achsen (keine implizite Ableitung aus Kalenderdaten):

- **`supportLifecycle`:** `CURRENT` (primäre Linie) | `SUPPORTED` (voll unterstützt) | `DEPRECATED` (Nachfolger empfohlen) | `RETIRED` (beendet; erzwingt `UNPUBLISHED` + `BLOCK`).
- **`publicationStatus`:** `DRAFT` | `PUBLISHED` | `UNPUBLISHED`.
- **`newSessionPolicy`:** `ALLOW` (normaler Start) | `WARN` (erfordert veröffentlichten, geprüften nicht-null `successor` auf First-Party-Allowlist und ausdrückliche Nutzerwahl) | `BLOCK` (keine Session/Capability; mit geprüftem Nachfolger `MAJOR_UPGRADE_REQUIRED` als identifierfreier Handoff ohne Token-/ID-Übernahme, mit `successor: null TEMPORARILY_UNAVAILABLE`). `policyRevision` steigt monoton und invalidiert ältere Start-Capabilities sofort.

```
{
  "$schema: "https://json-schema.org/draft/2020-12/schema",
  "$id: "https://skillpilot.com/schemas/openai/contract-line/v1",
  "type: öbject",
  ädditionalProperties: false,
  "required: [
    "contractMajor",
    "policyRevision",
    "displayName",
    ßupportLifecycle",
    "publicationStatus",
    "newSessionPolicy",
    ßuccessor"
  ]
}
```

(Vollständige Feld- und `allOf`-Bedingungen für `WARN -> successor != null` und `RETIRED -> UNPUBLISHED + BLOCK` werden durch die Release- und CI-Validierung gegen `$id` geprüft; die aktuelle V1-Konfiguration steht nach dem Web-first-Schnitt auf `policyRevision: 3`, `CURRENT`, `DRAFT`, `ALLOW`, `successor: null`.)

- **Verbindliche CI/CD-Gates:** (1) Versions-, Namens- und Manifestgrenzen-Konsistenz, (2) automatischer `Contract-Diff` gegen veröffentlichte Snapshots (blockiert entfernte Tools/Felder, neue Pflichtfelder, Typ-/Enum-/Security-Brüche und veränderte unveränderliche UI-URIs), (3) `Skill-Contract-Lint` und reproduzierbare Bundle-Hashes, (4) `Contract-Testmatrix` aller nicht-`RETIRED` Linien samt Auth-Audience/Scope- und UI-Integritätstests sowie (5) Zustandsmigrations- und Rollbacktests.
- **Observability & Log-Datenschutz:** Protokolliert werden `contractMajor`, `pluginLine`, `serverBuild`, `toolName`, `status`, stabiler `resultCode`, Latenz sowie (sofern im versionierten Resultat vorhanden) `stateVersion`, `stateSchemaVersion`, `workflowVersion`, `curriculum Revision` und bei Writes `clientRequestId`. Die `learningSessionId` erscheint ausschließlich

als gekürzter HMAC-SHA-256-Fingerprint und niemals im Klartext; hochkardinale Korrelatoren und Versionen sind reine Logfelder, während Metrik-Tags strikt auf `contractMajor`, `pluginLine`, `toolName`, `status` und `resultCode` begrenzt bleiben.

- **Historische Einordnung des abgelösten Mehr-Plugin-Linienmodells:** Der frühere Plan vom Juli–September 2026 sah vor, jede spätere entfernbare Major-Generation als eigene Plugin-Identität (`skillpilot-coach-v1..v9` auf `mcp-coach-v1..v9.skillpilot.com`, vgl. OpenAI-Forum-Anfrage vom 31. Juli 2026) mit eigenem Parallelbetrieb, optionaler expliziter Sessionmigration (`mig_...`) und späterem Unpublish/Delete zu führen; nie veröffentlichte Sprachhosts (`mcp-coach-de-v*/en-v*`) wurden bereits damals verworfen. Mit der Entscheidung vom 19. Oktober 2026 (10.1) bleibt die saubere interne Adaptertrennung (`v1`, später optional `v2` im selben `skillpilot-server`) erhalten, während nach außen dauerhaft genau **eine** Plugin-Identität (`skillpilot-coach` auf `https://mcp-openai.skillpilot.com/mcp`) geführt wird.