smartlearn
virtual learning. real success.

AIB_2026_Serie_0_Challenge-03_KI-Agent bei MedFinder

ICT-Berufsbildung Schweiz

Inhaltsverzeichnis

Ausgangslage **3**

Aufgabe **4**
Teilaufgabe a) Integrationsanforderungen an die Zwischenschicht 5
Teilaufgabe b) Konkrete Treiber (Kennzahlen, Systemeigenschaften, Einschränkungen) 7
Teilaufgabe c) Datenklassifikation und Datenbankempfehlung 8

Virtuelle Umgebung **9**
Installation, Setup und Konfiguration 9
Ubuntu Server 9
Upload der Applikation auf VM 10
Windows 11 Client 12

Ausgangslage

Ausgangslage

Ein mittelgrosses Unternehmen betreibt eine schweizweite Plattform zur Vermittlung von Ärzteschaft und Kliniken. Das Verzeichnis umfasst 28'000 Einträge in vier Landessprachen. Trotz vollständigem Datenbestand verlassen viele Nutzende die Mobile-App ohne Ergebnis: Die klassische Texteingabe verzeiht keine Tippfehler und setzt voraus, dass Namen und Fachbegriffe korrekt bekannt sind.

Das Unternehmen hat entschieden, diesem Problem mit einem sprachgesteuerten KI-Agenten zu begegnen. Nutzende sollen ihre Anfrage künftig einfach aussprechen können; so wie sie denken: *«Ich habe Rückenschmerzen – empfehle mir bitte einen Arzt in Zürich».*

Der Agent versteht die Absicht, gleicht sie mit dem Verzeichnis ab und liefert eine direkte, gefilterte Antwort. Zwischen dem Legacy-System und dem KI-Agenten wird dazu eine neue Zwischenschicht (Middleware) eingeführt.

Technische Rahmenbedingungen:

Datenschnittstelle: Der einzige verfügbare Zugang zum bestehenden Backend (`GET /v1/providers/dump`) liefert stets den vollständigen Datenbestand als eine einzelne JSON-Datei (383 MB). In der aktuellen Betriebsumgebung liegt die beobachtete Antwortzeit typischerweise im Bereich von 40-90 Sekunden, abhängig von Netzwerkauslastung und Serverlast. Filter- oder Paginierungsparameter stehen nicht zur Verfügung. Eine Anpassung des Endpoints vor Go-Live ist ausgeschlossen.

Spracherkennung: Empirische Untersuchungen auf Basis von 200 Deutschschweizer Sprachaufnahmen zeigen eine Fehlerquote von 23 % bei Fachpersonen-Namen und 34 % bei mundartlichen Ortsnamen. Phonetische Indexfelder fehlen im Datenbestand vollständig. Die Zwischenschicht muss fehlerhafte Transkriptionen dennoch korrekt auflösen können.

Architektur: Der KI-Agent kommuniziert ausschliesslich über REST-APIs mit standardisierten JSON-Antworten. Ein direkter Zugriff auf Datenbanken oder interne Systeme ist architekturbedingt nicht vorgesehen. Die Zwischenschicht übernimmt die Rolle der einzigen Schnittstelle zwischen Legacy-Backend und dem Agenten.

Umgebung und Zugangsdaten

Funktion	Host	IP-Adresse	OS	Applikationen & Werkzeuge	Benutzer	Passw
Legacy-API-Server	ubu-srv-exam	192.168.110.20/24	Ubuntu 24.04.2 LTS	Provider Directory API Swagger UI: http://192.168.110.20/docs (http://192.168.110.20/docs) Backend (Port 80): http://192.168.110.20/v1/providers/dump (http://192.168.110.20/v1/providers/dump)	kein Zugriff	
Testclient	win-cli-exam	192.168.110.101/24	Windows 11 Client	Browser Edge curl 8.19.0 Postman 12.11.3	vmadmin	sml1

Aufgabe

Im bereitgestellten API-Playground steht der Legacy-Endpoint GET `/v1/providers/dump` zur Verfügung. In der unten stehenden Tabelle sind die Zugangsdaten des Testclients zu entnehmen. Rufen Sie über den Testclient den Endpoint auf und studieren Sie den zurückgegebenen Datenausschnitt sorgfältig.

Umgebung und Zugangsdaten

Funktion	Host	IP-Adresse	OS	Applikationen & Werkzeuge	Benutzer	Passw
Legacy-API-Server	`ubu-srv-exam`	`192.168.110.20/24`	Ubuntu 24.04.2 LTS	Provider Directory API Swagger UI: http://192.168.110.20/docs (http://192.168.110.20/docs) Backend (Port 80): http://192.168.110.20/v1/providers/dump (http://192.168.110.20/v1/providers/dump)	kein Zugriff	
Testclient	`win-cli-exam`	`192.168.110.zzz/24`	Windows 11 Client	Browser Edge curl 8.19.0 Postman 12.11.3	`vmadmin`	`sml1`

Teilaufgabe a) Integrationsanforderungen an die Zwischenschicht

Analysieren Sie die Ausgangslage sowie die über den Endpoint abgerufenen Beispieldaten. Leiten Sie daraus vier zentrale Intergrationsanforderungen an die neue Zwischenschicht ab. Je eine pro Kategorie:
- Latenz/Performance
- Datenqualität/Suche
- Technische Kompatibilität

- Transkriptionsresilienz

Formulieren Sie jede Anforderung als präzise, technisch begründete Muss-Anforderung nach folgendem Muster:
Dimension: [Bezeichnung]
Anforderung: Die Zwischenschicht muss….
Begründung: Weil …. [Bezug zur Ausgangslage oder den API-Daten]

Punkteverteilung: 4 × 1 Punkt = 4 Punkte
Je eine vollständige Muss-Anforderung pro Dimension (Anforderung + Begründung). Ein Punkt pro Dimension nur wenn alle beideTeile vorhanden und kohärent sind. Teilpunkte werden nicht vergeben.

Anforderung an Latenz/Performance

Anforderungen an Suche

Anforderung an Technische Kompatibiltät

Anforderung an Transkriptionsresilienz

Teilaufgabe b) Konkrete Treiber (Kennzahlen, Systemeigenschaften, Einschränkungen)

Worauf stützen Sie Ihre Anforderungen?
Benennen Sie für jede der vier Anforderungen (Latenz/Performance; Datenqualität/Suche; Technische Kompatibilität; Transkriptionsresilienz) den konkreten Treiber aus der Ausgangslage oder den Beispieldaten: Eine Kennzahl, eine Systemeigenschaft oder eine technische Einschränkung, die zeigt, warum diese Anforderung unumgänglich ist.

Verwenden Sie folgendes Antwortschema:
Dimension: [Bezeichnung]

Treiber: [Konkrete Zahl, Feldname oder Eigenschaft aus der Ausgangslage/die Beispieldaten]

Erläuterung: Dieser Wert/Diese Eigenschaft macht die Anforderung unmöglich, weil….

Punkteverteilung: 4 × 1 Punkt = 4 Punkte
Pro Dimension 1 Punkt. **Wichtig:** Blosse Wiederholung der Anforderung aus Teilaufgabe a) = 0 Punkte. Es muss ein konkreter Datenpunkt aus der Ausgangslage oder den API-Beispieldaten genannt werden.

Auslöser Latenz/Performance

Auslöser Suche

Auslöser Technische Kompatibilität

Auslöser Transkriptionsresilienz

Teilaufgabe c) Datenklassifikation und Datenbankempfehlung

Bestimmen Sie anhand der Beispieldaten den Datentyp, im Sinne der Datenklassifikation und begründen Sie in 1-2 Sätzen, welche Art von Datenbank für die Zwischenschicht geeignet ist.

Punkteverteilung:
1 Punkt Datentyp + Klassifikation, 1 Punkt Datenbankbegründung

Datentyp

Datenbank Typ

Virtuelle Umgebung

Es werden VM aus dem Set `Master-Set PDS 2026` verwendet, weil diese bereits hilfreiche Konfigurationen und Skripts enthalten (die leider im Moment an anderer Stelle beschrieben sind).

Die virtuelle Umgebung umfasst folgende VM

- `192.168.110.2`:- Gateway - **bsd-inf-exam** - für Kandidaten unsichtbar (-)
- `192.168.110.20` - Ubuntu Server - **ubu-srv-exam** - für Kandidaten nur sichtbar, aber nicht steuerbar (R)
- `192.168.110.101` - Windows 11 Client - **win-cli-exam** - für Kandidaten sicht- und steuerbar (RW)

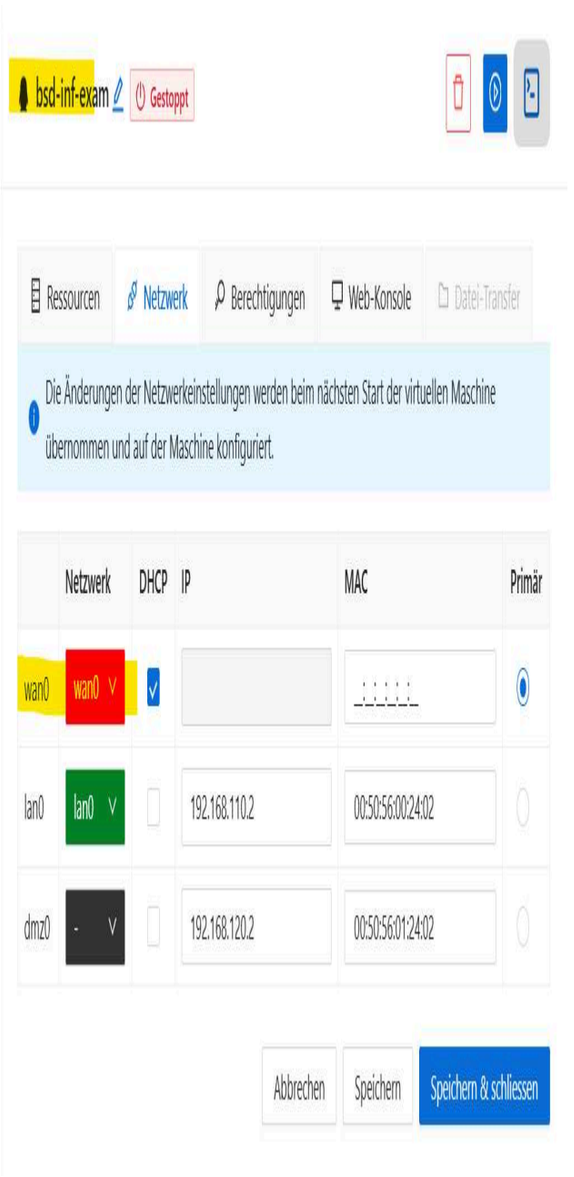
Installation, Setup und Konfiguration

Ubuntu Server

- Bugfix wegen Hänger auf eth1 bei Booten
- Updates mit Standardskript
- Konfiguration der Locales (Sprachen und Tastaturlayouts) mit Standardskript
- Installation Docker
    - `curl -fsSL https://get.docker.com | sudo sh && sudo usermod -aG docker $USER && newgrp docker`
    - Test: `sudo systemctl status docker`
    - `docker compose version`
- Anpassungen docker-compose.yml
    - Binding auf NIC (nicht 127.0.0.1) und Port 80: `"80:8080"`
- Berechtigungen auf Verzeichnis `data`: `sudo chmod 777 data` (Hint: Container läuft mit vmadmin als non-root User)
- Build und Start der Umgebung: `docker compose down && sudo rm -rf data/* && docker compose up --build`

Upload der Applikation auf VM

- Für den Build des Dockers muss Internet vorhanden sein.
- Interface `wan0` auf Gateway `bsd-inf-exam` aktivieren und VM des Gateways starten.



- VM des Ubuntu Servers starten
- Upload der gezippten Applikation mit docker-compose.yaml auf die VM via Datei-Transfer



- Verbindung auf Server mittels Remote-Console
- Login als `vmdamin` mit Standardpasswort
- Wechsel auf `sudo -i`
- Entpacken der Applikation im Verzeichnis `/home/vmadmin/app`
- Anpassungen in `docker-compose.yml` bezüglich Binding: NIC (nicht 127.0.0.1) und Port 80: `"80:8080"`
- Build der App mittels `docker compose build`
- Docker mittels `docker compose up -d` (oder Server neu starten, Docker-Service startet App automatisch)
- Testen auf Client mit und ohne Internet
- Internetverbindung auf Gateway schliessen (VM muss gestoppt werden)

Windows 11 Client

- Updates mit Standardskript
- Konfiguration der Locales (Sprachen und Tastaturlayouts) mit Standardskript
- Bookmarks und Startseite im Edge
    - http://192.168.110.20/docs (http://192.168.110.20/docs)
    - http://192.168.110.20/v1/providers/dump (http://192.168.110.20/v1/providers/dump)
- Download und Installation Postman (als Admin), Quelle: https://www.postman.com/downloads/ (https://www.postman.com/downloads/)