smartlearn
virtual learning. real success.

Challenge-03_Agent d'IA pour MedFinder

ICT-Formation professionnelle Suisse

Table des matières

Situation initiale **3**

Mandat **5**

Mandat a) Exigences d'intégration relatives à la couche intermédiaire 6

Mandat b) Facteurs déterminants concrets (valeurs chiffrées, caractéristiques du système, contraintes) 7

Mandat c) Classification des données et recommandation d'une base de données 9

Environnement virtuel **9**

Installation, setup et configuration 10

Ubuntu Server 10

Upload de l'application sur la machine virtuelle 10

Windows 11 Client 13

Situation initiale

Situation initiale

Une entreprise de taille moyenne exploite une plateforme suisse de référencement de médecins et de cliniques. L'annuaire comprend 28 000 entrées réparties dans les quatre langues nationales. Malgré l'exhaustivité de la base de données, de nombreux utilisateurs quittent l'application mobile sans obtenir de résultat: la recherche textuelle classique ne tolère aucune faute de frappe et suppose que les noms et termes spécialisés soient connus avec précision.

L'entreprise a décidé de résoudre ce problème grâce à un agent d'IA à commande vocale. À l'avenir, les utilisateurs pourront formuler leur requête oralement, avec leurs propres mots: *«J'ai mal au dos – recommande-moi un médecin à Zurich.»*

L'agent d'IA comprend l'intention de l'utilisateur, la met en correspondance avec l'annuaire et renvoie une réponse directe et filtrée. À cet effet, une nouvelle couche intermédiaire (middleware) est introduite entre le système existant (legacy system) et l'agent d'IA.

Conditions techniques:

Interface de données: le seul accès disponible au backend existant (GET /v1/providers/dump) renvoie systématiquement l'ensemble des données sous la forme d'un unique fichier JSON (383 Mo). Dans l'environnement d'exploitation actuel, le temps de réponse observé se situe généralement entre 40 et 90 secondes en fonction de la charge du réseau et du serveur. Aucun paramètre de filtrage ni de pagination n'est disponible. Une modification du point de terminaison (endpoint) avant la mise en production est exclue.

Reconnaissance vocale: des études empiriques fondées sur 200 enregistrements vocaux en suisse allemand montrent un taux d'erreur de 23% pour les noms des professionnels de santé et de 34% pour les noms de localités prononcés en dialecte. Bien que le jeu de données ne comporte aucun champ d'indexation phonétique, la couche intermédiaire doit être capable de résoudre correctement les erreurs de transcription.

Architecture: l'agent d'IA communique exclusivement au moyen d'API REST renvoyant des réponses JSON standardisées. En raison de l'architecture du système, aucun accès direct aux bases de données ni aux systèmes internes n'est prévu. La couche intermédiaire constitue l'unique interface entre le backend existant (*legacy*) et l'agent.

Environnement et données d'accès

Fonction	Hôte	Adresse IP	OS	Applications et outils	Utilisateur	Mot de passe
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)	aucun accès	
Testclient	win-cli-					

exam	192.168.110.101/24	Windows 11 Client	Browser Edge curl 8.19.0 Postman 12.11.3	vmadmin	sml12345

Mandat

Le point de terminaison hérité GET /v1/providers/dump est disponible dans l'API Playground fournie. Les données d'accès du client de test figurent dans le tableau ci-dessous. Appelez le point de terminaison à l'aide du client de test et étudiez attentivement l'extrait de données renvoyé.

Environnement et données d'accès

Fonction	Hôte	Adresse IP	OS	Applications et outils	Utilisateur	Mot de passe
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)	aucun accès	
Testclient	win-cli-					

exam	192.168.110.zzz/24	Windows 11 Client	vmadmin	sml12345

Browser
Edge

curl 8.19.0

Postman
12.11.3

Mandat a) Exigences d'intégration relatives à la couche intermédiaire

Analysez la situation initiale ainsi que les données d’exemple récupérées via le point de terminaison. Sur la base de cette analyse, déduisez quatre exigences d’intégration essentielles pour la nouvelle couche intermédiaire, à raison d’une exigence pour chacune des catégories suivantes:
- Latence/Performance
- Qualité des données/Recherche
- Compatibilité technique

- Résilience aux erreurs de transcription

Formulez chaque exigence sous la forme d’une exigence impérative (« doit ») précise et justifiée techniquement, selon le modèle suivant:
Catégorie: [intitulé]
Exigence: La couche intermédiaire doit…
Justification: parce que… [référence à la situation initiale ou aux données de l’API]

Attribution des points: 4 × 1 point = 4 points
Un point par catégorie est attribué uniquement si une exigence impérative complète (exigence + justification) est formulée et que les deux éléments sont cohérents. Aucun point partiel n’est accordé.

Exigence relative à la latence/performance

Exigence relative à la qualité des données/recherche

Exigence relative à la compatibilité technique

Exigence relative à la résilience aux erreurs de transcription

Mandat b) Facteurs déterminants concrets (valeurs chiffrées, caractéristiques du système, contraintes)

Sur quels éléments fondez-vous vos exigences?
Pour chacune des quatre exigences (Latence/Performance; Qualité des données/Recherche; Compatibilité technique; Résilience aux erreurs de transcription), indiquez le motif concret issu de la situation initiale ou des données d'exemple qui rend cette exigence indispensable: une valeur chiffrée, une

Utilisez le modèle de réponse suivant:
Catégorie: [intitulé]

Facteur déterminant concret: [valeur chiffrée, nom de champ ou caractéristique issue de la situation initiale ou des données d'exemple]

Explication: Cette valeur/caractéristique rend le respect de cette exigence difficile voire impossible parce que…

Attribution des points: 4 × 1 point = 4 points
1 point par catégorie. **Important:** la simple répétition de l'exigence formulée dans la question a) = 0 point. Un élément concret issu des données de la situation initiale ou des données d'exemple de l'API doit être cité.

Facteur déterminant Latence/Performance

Facteur déterminant Qualité des données/Recherche

Facteur déteminant Compatibilité technique

Facteur déterminant Résilience aux erreurs de transcription

Mandat c) Classification des données et recommandation d'une base de données

Sur la base des données d’exemple, déterminez le type de données selon la classification des données et justifiez en 1 à 2 phrases quel type de base de données est approprié pour la couche intermédiaire.

Attribution des points:
1 point pour le type de données + classification, 1 point pour la justification de la base de données

Type de données

Type de base de données

Environnement virtuel

Les machines virtuelles (VM) de l’environnement `Master-Set PDS 2026` sont utilisées, car elles contiennent déjà des configurations et des scripts utiles (malheureusement documentés ailleurs pour le moment).

L’environnement virtuel comprend les VM suivantes:

- `192.168.110.2`:- Gateway - **bsd-inf-exam** - invisible pour les candidats (-)
- `192.168.110.20` - Ubuntu Server - **ubu-srv-exam** - visible pour les candidats, mais non contrôlable (R)
- `192.168.110.101` - Windows 11 Client - **win-cli-exam** - visible et contrôlable par les candidats (RW)

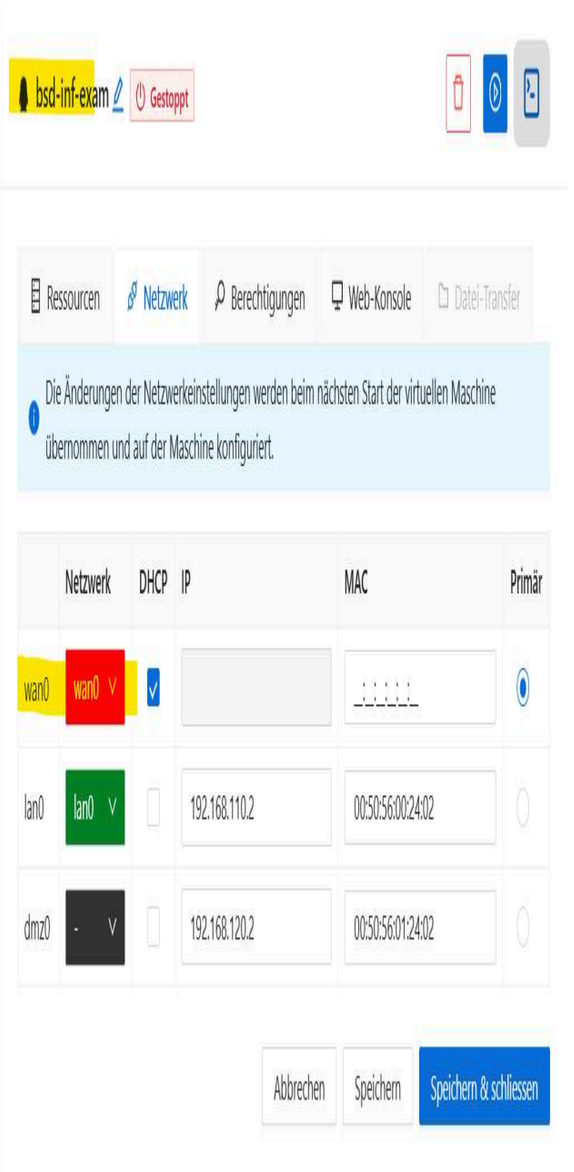
Installation, setup et configuration

Ubuntu Server

- Correction du bug lié au blocage sur eth1 lors du démarrage
- Mises à jour avec le script standard
- Configuration des locales (langues et dispositions de clavier) avec le script standard
- 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`
- Adaptations docker-compose.yml
    - Binding sur la NIC (et non sur 127.0.0.1) port 80: `"80:8080"`
- Droits d'accès au répertoire `data`: `sudo chmod 777 data` (indication: le conteneur s’exécute avec l’utilisateur non root vmadmin)
- Construction et démarrage de l’environnement: `docker compose down && sudo rm -rf data/* && docker compose up --build`

Upload de l'application sur la machine virtuelle

- Une connexion Internet est requise pour construire l'image Docker
- Activer l'interface `wan0` sur la passerelle `bsd-inf-exam`, puis démarrer la VM de la passerelle



- Démarrer ensuite la machine virtuelle du serveur Ubuntu
- Téléverser sur la VM l'application compressée (ZIP), accompagnée du fichier docker-compose.yml, à l'aide du transfert de fichiers



- Se connecter au serveur via la console distante
- Se connecter en tant qu'utilisateur `vmdamin` à l'aide du mot de passe par défaut
- Passer en mode `sudo -i`
- Extraire l'application dans le répertoire `/home/vmadmin/app`
- Adapter le fichier `docker-compose.yml` concernant le binding: NIC (et non 127.0.0.1) et port 80: `"80:8080"`
- Construire l'image Docker de l'application à l'aide de `docker compose build`
- Démarrer le conteneur Docker avec `docker compose up -d` (ou redémarrer le serveur; le service Docker lance automatiquement l'application au démarrage)
- Tester l'application depuis le client, avec puis sans connexion Internet
- Désactiver la connexion Internet sur la passerelle (la VM doit être arrêtée)

Windows 11 Client

- Mises à jour à l’aide du script standard
- Configuration des locales (langues et dispositions de clavier) à l'aide du script standard
- Configurer les favoris et la page d’accueil dans 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)
- Téléchargement et installation de Postman (en tant qu’administrateur), source: https://www.postman.com/downloads/ (https://www.postman.com/downloads/)