Stručni zadatak za Python Developera – *TicketHub*

Uvod

Tijekom AI Akademije razvijat ćete agentni AI sustav za podršku u obradi korisničkih zahtjeva. Kao prvi korak potreban nam je *middleware* REST servis – **TicketHub** – koji prikuplja i izlaže „support tickete" iz (trenutačno samo jednog) vanjskog izvora.

Cilj zadatka je procijeniti vaše razumijevanje **FastAPI**-ja, asinkronog programiranja u Pythonu te dobre inženjerske prakse (testovi, CI/CD, dokumentacija).

Tehnološki stack (minimalna očekivanja)

Tehnologija	Verzija	Napomene
Python	3.11	Koristite typing, async/await
FastAPI	0.111	Automatski OpenAPI opis
httpx	0.27	Za pozive prema vanjskim servisima
pydantic	2.7	Validacija i serializacija
pytest	–	Jedin. i integr. testovi

Nice to have: SQLAlchemy + SQLite/PostgreSQL, Redis (caching), Docker Compose.

Vanjski izvor podataka

Koristite DummyJSON REST servis:

- Ticketi: https://dummyjson.com/todos
- Korisnici: https://dummyjson.com/users

Transformirajte polja u vlastiti **Ticket** model:

- id: *int*
- title: preuzeto iz polja todo
- status: "closed" ako je completed == true, inače "open"
- priority: izračunajte (npr. id % 3 → low/medium/high)
- assignee: korisničko ime preuzeto preko userId

Zadaci

Implementirajte sljedeće **GET** endpointove:

1. /tickets – paginirana lista (id, title, status, priority, opis ≤ 100 znakova)
2. /tickets/{id} – detalji ticketa + puni JSON iz izvora
3. /tickets?status=<>&priority=<> – filtriranje
4. /tickets/search?q=… – pretraga po nazivu

Nice to have:

- /stats – agregirane statistike
- Autentifikacija (JWT) pomoću /auth/login DummyJSON-a
- Caching (Redis ili in-memory TTL)
- Rate limiting (npr. slowapi)
- Logiranje (INFO/WARNING/ERROR)
- Health-check endpoint za k8s/Compose

Dodatni zahtjevi

- Jasna struktura projekta (src/, tests/, ci/) + PEP-8 stil
- CI workflow (GitHub Actions ili slično)
- README.md s uputama:
    - postavljanje okruženja
    - konfiguracija varijabli
    - Makefile/taskfile za run/lint/test/docker-build
    - Dockerfile + docker-compose.yml (uklj. Redis ako treba)
- Komitovi trebaju biti feature-based i jasno imenovani
- *Bonus:* statička HTML dokumentacija OpenAPI-ja (npr. redoc-static)

Kako predati rješenje

1. Forkajte privatni GitHub repozitorij i pošaljite nam link
2. Kreirajte feature branch solution/<ime_prezime>
3. Nakon dovršetka izradite merge request i obavijestite nas

Napomena

Slobodno koristite **ChatGPT** ili **Gemini** – napišite gdje i zašto ste ih koristili (komentar u kodu ili README). Bonus ako priložite prompt.

Sretno i vidimo se na intervjuu!