Milestone 1 – Alegerea proiectului

Platformă distribuită tip LeetCode

Nume: Denis Covei
Grupa: 342C1

1. Arhitectura soluției

Platforma este implementată ca un sistem distribuit de microservicii, orchestrate cu Docker Swarm. Structura include:

- **Keycloak** – SSO cu OAuth2/OIDC; gestionează autentificarea și metadatele utilizatorilor.
- **User Service** – microserviciu propriu pentru gestionarea utilizatorilor și rolurilor.
- **Database Service (PostgreSQL)** – păstrează utilizatorii, problemele și submisiile.
- **Problem Management Service** – gestionează problemele și directoarele în care sunt stocate testele.
- **Code Runner Service** – execută codul utilizatorilor în containere izolate (sandbox) utilizând GCC.
- **Queue Service (RabbitMQ / Redis Streams)** – gestionează submisiile prin joburi *nepersistente*.
- **API Gateway (Traefik/Nginx)** – rutează traficul și oferă un punct unic de acces către microservicii.

Fluxul general:

1. Utilizatorul se autentifică prin Keycloak.
2. Trimite o soluție → Submission Service o scrie în DB.
3. Submission Service publică un job în coada de execuție.
4. Workerii Code Runner consumă jobul, rulează soluția pe testele problemei.
5. Statusul final este trimis înapoi în DB.

2. Funcționalități

- Login cu Keycloak (OpenID Connect).
- Crearea și administrarea utilizatorilor prin User Service.
- Managementul problemelor (creare, listare, modificare).
- Testele problemelor sunt stocate doar în filesystem (director per problemă).
- Trimiterea submisiilor prin API.
- Evaluarea asincronă a codului prin coadă de mesaje.
- Sandbox de execuție izolat pentru fiecare job.
- Rate limiting distribuit implementat cu Redis / Traefik.

3. Module avansate

Execuție sandbox distribuită cu replicare de workers

Mai multe containere Code Runner rulează în paralel și consumă joburile din coadă, oferind scalabilitate și latență redusă.

Rate-limiting distribuit

Limitează numărul de submisii pe utilizator într-un interval, folosind un mecanism distribuit (Redis sau API Gateway).

3.3. Baza de date (obligatoriu)

Persistența este asigurată printr-un volum Docker pentru PostgreSQL.

Baza de date conține **doar tabelele cerute**: users, problems, submissions.
Testele sunt salvate în **filesystem**, în volume Docker, nu în DB.

users

Coloană	Tip	Descriere
id	serial PK	identificator unic
username	text	nume de utilizator
email	text	adresă email
role	text	user / admin
created_at	timestamp	dată creare

problems

Coloană	Tip	Descriere
id	serial PK	identificator problemă
title	text	titlu problemă
description	text	enunțul problemei
tests_dir	text	calea către directorul cu testele problemei (ex: /problems/5/tests/)
created_at	timestamp	dată creare

Testele nu sunt în baza de date.
Fiecare problemă are un director cu:

- test1.in
- test1.out

- test2.in
- test2.out

  ... etc.

submissions

Coloană	Tip	Descriere
id	serial PK	identificator submit
user_id	int FK	utilizatorul care a trimis
problem_id	int FK	problema asociată
source_code	text	codul trimis
language	text	limbajul (C/C++)
status	text	rezultat final (success / CE / RE / TLE / etc.)
created_at	timestamp	dată trimitere