Platformă Distribuită pentru Organizarea Evenimentelor și Sincronizarea Calendarului de Grup

Alexandru Alin-Ioan
Grupa: 344C3

27 noiembrie 2025

Cuprins

1 Introducere

Această aplicție web propune o soluție pentru un calendar distribuit prin care utilizatorii să poată crea, edita și sincroniza evenimente în cadrul unor grupuri. Platforma va oferi funcționalitatea de gestionare a disponibilității tutror participanților și de găsire automată de intrevale libere pentru evenimentele planificate.

2 Obiectivele aplicației

- Oferirea unui calendar multi-utilizator cu sincronizare în timp real.
- Gestionarea disponibilității și a evenimentelor unui participant sau unui grup.
- Prevenirea conflictelor de rezervare prin mecanisme de distributed locking.
- Generarea de propuneri automate de meeting folosind procesare asincronă și workeri replicați.
- Implementarea unor mecanisme robuste de autentificare, autorizare și management al rolurilor folosind Keycloak.
- Livrarea proiectului sub forma unui set de servicii Docker.

3 Modulele aplicației

1. **Autentificare** – funcție de autorizare utilizând o tehnologie de Single Sign-On (SSO).
2. **Profil utilizator + roluri** – Keycloak pentru managementul utilizatorilor, rolurilor și token-urilor.
3. **Baza de date** – gestionată prin ORM.
4. **Conflict resolution + Distributed Locking** – gestionează situațiile concurente în care mai mulți utilizatori creează sau modifică evenimente în același interval temporal.
5. **Suggestion Worker replicat** – calculează automat intervalele comune libere pentru un grup. Joburile sunt transmise de către serviciul Calendar către broker, care mai apoi le redistribuie către workeri.

4 Descrierea arhitecturii

4.1 Componente principale

4.1.1 Serviciul de autentificare (Keycloak)

Folosește OAuth2 și emite token-uri JWT. Creează noi utilizatori, roluri și trimit aceste informații către serviciul de profil utilizatori

4.1.2 Serviciul de profil utilizatori

Preia informațile oferite de Keycloak și creează profiluri locale în baza de date. Se vor oferi 3 tipuri de utilizatori:

1. **Participant** – Are acces la calendarul grupului și își poate seta disponibiltatea.

2. **Organizer** – Poate planifica evenimente în funcție de disponibiltatea participanților din grup, să adauge sau să elimine utilzatori din grup și să desemneze alți utilizatori ca organizatori.

3. **Admin** – Are acces la toate utilitatile administrative pentru alți utilizatori.

4.1.3 Baza de date (PostgreSQL)

Baza de date va fi prezntă în două instanțe cu roluri diferite: Prima instanță va păstra informațile despre utilizatori și rolurile acestora, iar cea de a doua va reține informații despre calendar, intervalele ocpuate și evenimentele prezente.

4.1.4 Serviciul Calendar - replicat

Expune un API pentru managementul grupului, operații CRUD pentru evenimente și verificarea disponibilității. Serviciul Calendar este replicat pentru a gestiona un număr mare de cereri simultane. Deoarece logica sa este stateless, iar mecanismele de sincronizare sunt gestionate extern prin Redis (locking) și RabbitMQ (comunicare asincronă), replicarea nu introduce probleme de consistență și îmbunătățește disponibilitatea sistemului.

4.1.5 Message Broker (RabbitMQ)

Serviciu replicat care preia joburi de procesare de intrevale libere de la serviciul calendar și le trimite către un set de workeri proprii.

4.1.6 Serviciul Suggestion Worker - replicat

Serviciul Suggestion Worker este un modul distribuit care efectuează procesare paralelă asupra datelor din baza de date. Preia joburi de la broker și procesează în paralel cererile de găsire a unui interval comun pentru un grup. Un job conține informații unui grup și a unui interval (câteva ore sau zile), iar răspunsul va fi o listă cu intervale de o oră disponibile pentru toți membrii grupului în perioada specificată. Un worker va prelua disponibilitatea fiecărui membru al grupului în intervalul specificat de job și va realiza intersecția acestora pentru a găsi posibile locuri libere.

4.1.7 Serviciul de locking distribuit (Redis)

Gestionează mecanismele de *distributed locking* necesare evitării conflictelor la nivel de interval orar prin algoritmul *RedLock*. Mecanismul de locking distribuit este utilizat exclusiv de serviciul Calendar pentru a proteja operațiile critice de scriere asupra intervalelor orare și a preveni conflictele atunci când mai mulți utilizatori sau instanțe ale API-ului încearcă să modifice simultan aceleași date.

4.2 Rețelele

4.2.1 Rețeaua `frontend-net`

- Serviciul Calendar (replicat)
- Serviciul de autentificare (Keycloak)

4.2.2 Rețeaua `calendar-net`

- Serviciul Calendar (replicat)
- Message Broker (RabbitMQ)
- Baza de date PostgreSQL (instanța pentru calendar)
- Redis (locking distribuit)

4.2.3 Rețeaua `profile-net`

- Serviciul Calendar (replicat)
- Serviciul de profil utilizatori
- Baza de date PostgreSQL (instanța pentru profiluri)

4.2.4 Rețeaua `worker-net`

- Message Broker (RabbitMQ)
- Suggestion Worker (replicat)
- PostgreSQL (instanța pentru calendar)

5 Diagrama arhitecturii

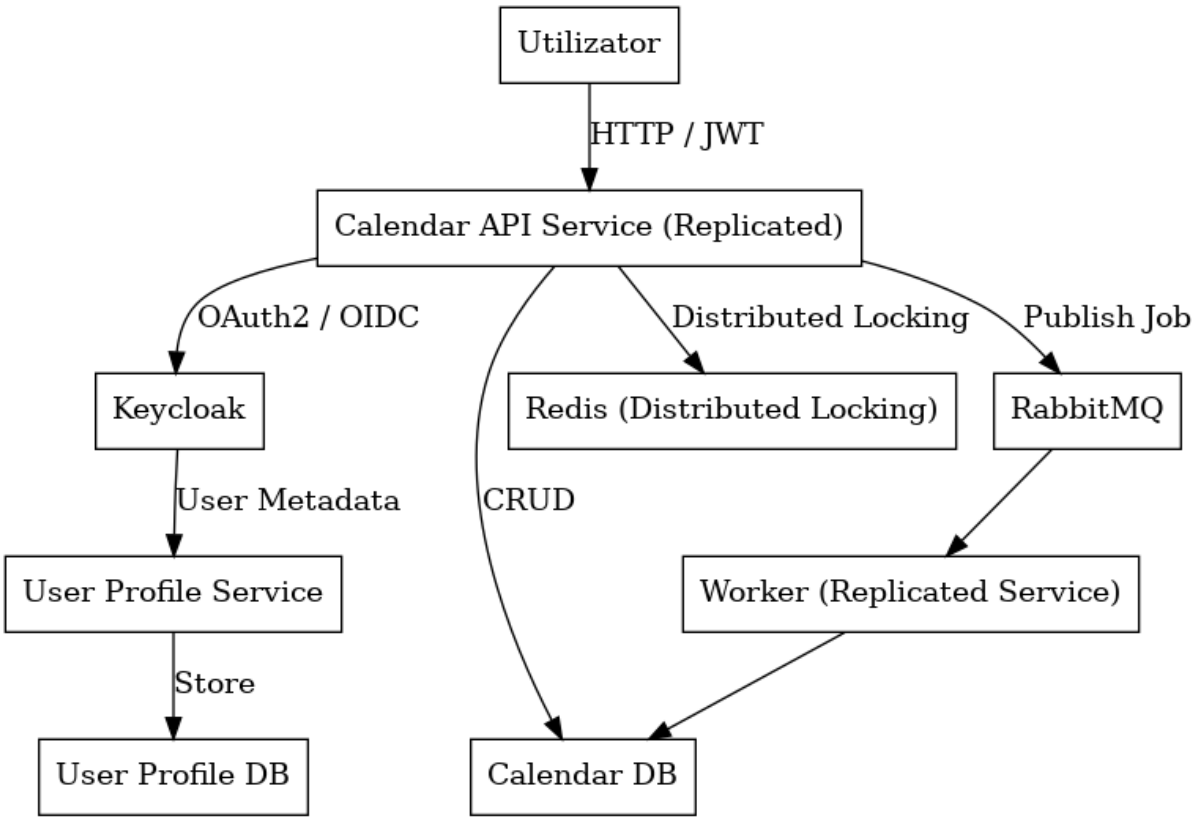

Figura 1: Arhitectura platformei de calendar distribuit