# PROBLÉMA-ANALÍZIS ENTERPRISE RENDSZEREKBEN

6. forduló

adnovum

A kategória támogatója: Adnovum Hungary Kft.

**RENDELKEZÉSRE ÁLLÓ IDŐ:**

15:00

## Ismertető a feladathoz

**Felhasznált idő:** 01:48/15:00 **Elért pontszám:** 0/32

## 1. feladat 0/10 pont

Mi fog történni: Van egy alkalmazásunk amiben a db connection pool mérete 10 és 1 db metódust publikál ami két egymásba ágyazott tranzakciót indit a meghívásakor (ezzel max 2 db connectiont használva egyidejüleg).

-----metódus eleje--->

-----db kapcsolat 1 eleje-->

-----db kapcsolat 2 eleje-->

-----db kapcsolat 2 vége-->

-----db kapcsolat 1 vége-->

-----metódus vége--->

A metódus futási ideje 1mp és a db connection timeout 5 másodperc. Mi történhet legrosszabb esetben amikor elérjük a 6 valamint később a 10 párhuzamos hivást másodpercenként?

**Válaszok**

- [x] Semmi, a rendszer elbirja a terhelést.
- [ ] 6 kérés müködik viszont 10 kérésnél holtpont alakul ki amiből nem lehet visszatérni.
- [ ] 6 kérés müködik viszont 10 kérésnél holtpont alakul ki amiből 5 másodperc után visszaáll a rendszer.
- [x] 6 kérésnél erőforrás kiéheztetés alakul ki amiből 1 másodperc után visszaáll a rendszer, 10 kérésnél holtpont alakul ki amiből 5 másodperc után lehet visszatérni.
- [ ] 6 kérésnél erőforrás kiéheztetés alakul ki amiből 5 másodperc után visszaáll a rendszer, 10 kérésnél holtpont alakul ki amiből nem lehet visszatérni.
- [ ] 6 kérésnél erőforrás kiéheztetés alakul ki amiből 1 másodperc után visszaáll a rendszer, 10 kérésnél holtpont alakul ki amiből nem lehet visszatérni.
- [ ] 6 kérésnél erőforrás kiéheztetés alakul ki amiből 1 másodperc után visszaáll a rendszer, 10 kérésnél holtpont alakul ki amiből 1 másodperc után lehet visszatérni.
- [ ] A rendszerben már 6 kérésnél deadlock alakul ki amiből 5 másodperc visszatérni.

**Magyarázat**

5 kérésnél 10 kapcsolat fogyhat el maximum. Ebben a pillanatban a 6. hivásnak max 1 másodpercet kell várnia hogy fölszabaduljon egy kapcsolat. 10 hivásnál előfordulhat hogy minden hivás csupán az első kapcsolatot szerzi meg és egymásra várnak a 2. kapcsolat fölszabadulásához ezért holtpont alakul ki ami 5 másodperc után a connection timeoutkor feloldódik.

## 2. feladat 0/8 pont

A rendszer információs pont kereső rendszere különböző hibák következtében gyakran leáll. A rendszer összetett keresési kifejezések alapján végzett számítások alapján jelenít meg találati listát a felhasználóknak. Sajnos a jelenlegi kialakítás mellett akár órák is eltelhetnek, mire a rendszer üzemeltetői észreveszik a problémát újraindítják a rendszert. Szerencsére az alkalmazásnak nem kell sok idő az inicializáláshoz, így újraindítás után szinte azonnal használatba vehetik a felhasználók.

Az alábbiak közül melyekkel érhetjük el, hogy a modul magasabb rendelkezésre állással működjön, és a fenti problémát kiküszöböljük?

**Válaszok**

- [x] monitorozzuk az infrastruktúra és az alkalmazás működését, probléma esetén automata riasztást küldünk

- [x] a komponenst képessé tesszük automatikus horizontális skálázódásra, és terhelés elosztón keresztül tesszük elérhetővé a többi modul számára
- [x] hiba esetén a felhasználókat átereljük egy standby szerverre
- [ ] hiba esetén a felhasználókat átereljük statikus HTML oldalra
- [x] Circuit breaker tervezési minta alkalmazásával

---

### Magyarázat

Az, hogy "hiba esetén a felhasználókat átereljük statikus HTML oldalra" is segít a felhasználói élmény növelésében hiba esetén, viszont egy dinamikus fórum modulnál a statikus HTML oldal nem elég.

## 3. feladat 0/14 pont

Az alkalmazásunk az RFC 7519 szabványnak megfelelő JWT tokent használ a felhasználók azonosítására. Az rendszer az alábbi kódrészlettel illusztrált módon autentikálja és autorizálja a felhasználókat:

```
Algorithm algo = Algorithm.RSA256(publicKey, privateKey);

String authenticate(String username, String password) {
    User user = userService.getUser(username, password);
    return JWT.create()
            .withIssuer("awesome-service")
            .withSubject(user.getEmail())
            .sign(algo);
}

String authorize(String token) {
    JWTVerifier jwtVerifier = JWT.require(algo)
            .withIssuer("awesome-service")
            .build();
    DecodedJWT jwt = jwtVerifier.verify(token);
    String currentUserEmail = jwt.getSubject();
    logger.info("Hello, " + currentUserEmail);
    return currentUserEmail;
}
```

A rendszer felhasználói natív Android és iOS alkalmazások, amelyek a bejelentkezés után a memóriájukban tárolják a felhasználó tokenjét.

Milyen lehetőségeink vannak, ha egy adott felhasználót szeretnénk biztonságosan kijelentkeztetni?

### Válaszok

- [ ] A kliens törli a tokenjét a saját memóriájából.
- [ ] A rendszert kiegészítjük egy "logout" metódussal, ami egy új "algo" objektumot hoz létre egy másik aláírókulccsal. Ha egy felhasználó ki szeretne jelentkezni, meghívja ezt a metódust.
- [ ] A rendszert kiegészítjük egy "logout" metódussal, ami egy tokent vár paraméterként és érvényteleníti azt. Visszatérési értéke az új, érvénytelenített token amit visszaküldünk a felhasználónak.
- [x] Az authenticate metódust módosítjuk, hogy rövid élettartamú tokeneket állítson elő, és kiegészítjük egy "refresh" metódussal, ami egy érvényes token alapján egy frisset ad vissza. A rendszerből való kijelentkezés úgy történik, hogy a felhasználó nem újítja meg a tokenét, és az aktuális tokent törli a saját memóriájából.
- [x] A rendszert kiegészítjük egy "logout" metódussal, ami a paraméterként átadott tokent elhelyezi egy globális tiltólistában. Emellett módosítjuk az "authorize" metódust, hogy csak azt a tokent engedélyezze, ami nincs a tiltólistán.

### Magyarázat

A JWT token nem módosítható és önleíró, ezért az alábbi válaszok helyesek:

A 4-es és az 5-ös válasz, a 4-es válasz azért, mert ez a lehetőség a rövid élettartammal és a folyamatos megújítással valósítja meg biztonságosan azt, hogy a nem aktív felhasználói munkamenetek érvénytelenítésre kerüljenek. Az 5-ös válasznál ez a fajta megoldás kiegészíti a JWT mechanizmust egy plusz tiltólistával, ami a JWT érvényességét felülírhatja.

A többi megoldás különböző okok miatt nem megfelelő:

Az 1-es választól a token még érvényes marad a szerver oldalon, illetve ha illetéktelenek kezébe jutott, azok korlátlan ideig felhasználhatják, mivel nem állítunk be lejáratot. A 2-es válaszban leírt metódussal ha valaki ki szeretne lépni, az összes többi felhasználó tokenjét is érvényteleníténénk. A 3-as válaszban az a helytelen, hogy a JWT token nem módosítható.

## 4. feladat 0/0 pont

Mit célokat tudunk elérni aszinkron kommunikáció segítségével?

### Válaszok

- [ ] A kliens gyorsabban kap válszt
- [x] Terhelhetőbb rendszert
- [x] Konkurenciakezelés
- [x] Több fogadó fél dolgozhatja föl az üzeneteket egyszerre
- [x] Késleltetett földolgozás

## Magyarázat

***Frissítés (2021.12.03):*** A kérdés nem kötötte ki, hogy kliensoldali aszinkronitásról van szó, ezért "A kliens gyorsabban kap választ" válaszlehetőség is helyes. Mivel így a kérdésre az összes válasz helyes lenne, ami a versenyszabályzat szerint nem lehetséges, 0 pontosra állítottuk a kérdést.