JAVA SPRING MICROSERVICES

1. forduló

A kategória támogatója: Vodafone Magyarország Zrt.

Ismertető a feladathoz

Kérjük, hogy a feladatlap indítása előtt mindenképp olvasd el az alábbi útmutatót:

- MINDEN kérdésre **van helyes válasz**.
- Olyan kérdés **NINCS**, amire az összes válasz helyes, ha mégis az összes választ bejelölöd, arra a feladatra automatikusan 0 pont jár.
- A **radio button-os kérdésekre egy helyes válasz van**.
- **Ha lejár a feladatlap ideje, a rendszer AUTOMATIKUSAN** beküldi azt az addig megjelölt válaszokkal.
- Azokat a feladatlapokat, amelyekhez **csatolmány** tartozik, javasoljuk **NEM mobilon** elindítani, erre az érintett feladatlapok előtt külön felhívjuk a figyelmet.
- Az **adatbekérős feladatokra NEM jár részpontszám**, csak a feleletválasztósakra.
- **Helyezéseket a 4. forduló után mutatunk**, százalékos formában: adott kategóriában a TOP 20-40-60%-hoz tartozol.
- **Badge-ket** szintén a 4.forduló után kapsz majd először.
- Ha egyszerre több böngészőből, több ablakban vagy több eszközről megnyitod ugyanazt a feladatlapot, **nem tudjuk vállalni** az adatmentéssel kapcsolatban esetlegesen felmerülő anomáliákért a felelősséget!
- A hét forduló során az egyes kategóriákban (de nem feltétlenül mindegyikben) **könnyű-közepes-nehéz kérdésekkel** egyaránt találkozhatsz majd.

Jó versenyzést kívánunk!

1.forduló: Nem spóroltam semmin! - John Hammond

Mr. Hammond a park következő generációján dolgozik és a csapatod azt a küldetést kapta, hogy készítsék el a parkot menedzselő és fenntartó rendszert egy új köntösben. Idő és pénz áll rendelkezésre, már csak elhivatottság és szakmai ismeretek szükségesek.

Mire lesz szükséged a fordulóban?

- Internet kapcsolat
- Java 11 telepítve
- Maven telepítve (vagy Maven Wrapper használata)
- Fejlesztői környezet (ajánlott)

Felhasznált idő: 00:00/40:00 **Elért pontszám:** 0/14

Indítás utáni csatolmányok

1. feladat 0/2 pont

Egy PoC (Proof of Concept) keretein belül a csapat pár tagja elkészítette az alkalmazás vázát, viszont keretrendszer nélkül valósították azt meg, de ahhoz, hogy a kódbázis stabil alapra épüljön emiatt a Spring Boot keretrendszert kell bevezetnetek, az eddigi követelmény alapján projektnek microservice alapú architektúrán kell megépülnie.

A következő felsorolásban lévő elemek melyike nem tartozik 12 faktoros alkalmazások elvébe?

Válaszok

- [ ] Függőségek explicit deklarálása csomagkezelő használatával
- [x] Teszt vezérelt fejlesztés követése
  Ez a válasz helyes, de nem jelölted meg.
- [ ] Közös kódbázis
- [x] Fájl vezérelt alkalmazás loggolás
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

A 12 faktoros alkalmazás elvek mint ahogy a neve is mondja, 12 faktort rögzít, ajánlásokat ad alkalmazások működésére. Microservice architektúrában ezek az elvek remek ajánlásokat adnak, hogyan is építsük és menedzseljük a szolgáltatásainkat a fejlesztési ciklusok és a szolgáltatások fejlesztés utáni intervallumában is.

A 12 faktoros alkalmazás elvek nem adnak javaslatot a Teszt vezérelt fejlesztésre, illetve a log kezelésről adnak, viszont ott az ajánlásuk az úgynevezett **standard output**-ra való írás és nem fájlba.

Hivatalos weboldal: https://12factor.net/

2. feladat 0/0 pont

A következő tulajdonságok közül melyek tartoznak a microservice architektúrára a szolgáltatás orientált architektúrával szemben?

Válaszok

- [ ] Globális / megosztott adatbázis minta
- [x] Adatbázis per szolgáltatás minta
  Ez a válasz helyes, de nem jelölted meg.
- [x] REST részesítése előnyben a SOAP-al szemben
  Ez a válasz helyes, de nem jelölted meg.
- [ ] A szolgáltatások egymás között az úgynevezett ESB-n keresztül kommunikálnak
- [x] A szolgáltatások lazán csatoltak
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

Kedves Versenyzők!

A kérdést 0 pontosra állítottuk, mivel az észrevételek alapján "A szolgáltatások lazán csatoltak" válaszlehetőség helyessége támadható a feladat értelmezése függvényében: ebben a megfogalmazásban ezt az állítást úgy is lehet érteni, hogy a SOA szolgáltatások nem lazán csatoltak, ami viszont nem igaz. Ezen kívül a magyarázatban megadott link nem hivatalos sztenderd, ami alapján kimondható, hogy mi számít microservice-nek.

A feladatíróval való egyeztetést követően az értelmezési kérdések miatt a fentiek mellett döntöttünk. (2022.10.24.)

Köszönjük a megértést!

A feladatírók és a szervezők

A SOA (szolgáltatás orientált architektúra) jópár pontban eltér a microservice architektúrától, de nem állnak annyira messzire egymástól. Microservicek esetén előnyben részesítik az egy adatbázis per szolgáltatás mintát, amely garantálja a szolgáltatások adat szintű elkülönítését is.

A HTTP protokollon belül a REST-et minősítik előnyben ilyen típusú architektúra használata közben a könnyedsége miatt, SOA-ban a SOAP részesített előnyben, de a kettő nem zárja ki egymást.

A microservice architektúrában a szolgáltatások lazán csatoltak.

Hivatalos weboldal: https://12factor.net/

3. feladat 0/2 pont

Milyen előnyei lehetnek a microservice alapú architektúrának?

Válaszok

- [x] A szolgáltatások függetlenül skálázhatóak lesznek egymástól.
  Ez a válasz helyes, de nem jelölted meg.
- [ ] A microservice architektúrában a csapatok csak egy megadott technológiát használhatnak, így nem kell keverednie a különböző programnyelveknek egy nagyobb projekten belül, ezzel tudják biztosítani, hogy az alkalmazás fejlesztése nem fog technológiai akadályokba ütközni.
- [ ] Könnyebb nagyobb fejlesztéseket végbevinni a szolgáltatásokon mert minden szolgáltatás csak az általa birtokolt üzleti domaint fedi le.
- [x] A CI/CD folyamatok leegyszerűsíthetők, az alkalmazást alkotó szolgáltatások külön adhatóak ki és telepíthetőek.
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

- A szolgáltatások skálázhatósága megnőhet, ha a rendszer megfelelően van kiépítve és bizonyos terhelési szituációk esetén csak azt a szolgáltatást kell skálázni amely éppen nagyobb nyomás alatt van. Egy monolit esetén ha az alkalmazás egy része nem megfelelően lett megtervezve/implementálva akkor nem biztos, hogy effektíven tudja kihasználni a rendelkezésre álló erőforrásokat, így egy esetleges felskálázás esetén sincs arra garancia, hogy az alkalmazás jobb teljesítménnyel fog rendelkezni.
- A microservice architektúra pontosan az ilyen technikai limitációkat törli el, a problémákra a "megfelelő" nyelvet és technológiát tudják használni a fejlesztők, a szolgáltatások megfelelői interfészeinek kidolgozása után bármilyen technológia használható.
- Ahogy a szolgáltatások száma növekszik egy ilyen architektúrában felépített projekten úgy vállhat egyre komplexebbé nagyobb fejlesztések kialakítása a kódbázisban, főleg ha azok több szolgáltatáson átívelnek, ezzel a fejlesztési, tesztelési és telepítési idő és komplexitás megnőhet.
- A CI/CD folyamatok egyszerűsödhetnek, ha egy szolgáltatásban hibát észlelünk és javítjuk akkor elég csak azt az egy szolgáltatást kitelepíteni a megfelelő környezetekre.

4. feladat 0/2 pont

Spring Boot projekt esetén milyen formában javasolt a függőségek megadása?

Válasz

- [ ] A függőségeket kézzel letöltjük és bemásoljuk a projekt könyvtárába.
- [x] A "**starter**" nevű függőségeket használjuk amelyeket a BOM (Bills of Materials) nevű függőségből érhetünk el.
  Ez a válasz helyes, de nem jelölted meg.

- ○ Felsoroljuk a Springes függőségeket és megadjuk hozzájuk a konkrét verziót.
- ○ Az "**extension**" nevű függőségeket használjuk amelyeket a BOM (Bill of Materials) nevű függőségből érhetünk el.

Magyarázat

- Kézzel egyáltalán nem ajánlott kezeli a függőségeket, a projekt mérete megnőhet a tárolóba feltöltött állományok miatt.
- A Spring Boot hivatalos dokumentációjában az úgynevezett **starter** nevű függőségeket ajánlják mert azok összegyűjtik a megfelelő verziójú tranzitív függőségeket.
- Kézzel felvenni Springes függőséget nem tiltott, de okozhat problémát ha a verziók nem kompatibilisek egymással.
- **extensions** nevű hivatalos függőségek nincsenek a BOM-ban, a Quarkus nevű keretrendszer használja ezt a formát.
- Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/using.html#using.build-systems.dependency-management

5. feladat 0/2 pont

Az alábbi lehetőségek közül hogyan aktiválhatjuk Spring Boot-on belül az úgynevezett **Auto Configuration** lehetőséget?

Válasz

- ○ @ComponentScan annotáció használatával
- ◉ @SpringBootApplication annotáció használatával
  Ez a válasz helyes, de nem jelölted meg.
- ○ @Configuration annotáció használatával

Magyarázat

- Spring Boot-on belül a legegyszerűbben a @SpringBootApplication nevű annotációval tudjuk ezt megtenni.
- A @ComponentScan annotáció segítségével csomagneveket adhatunk meg amely segítségével a csomagban lévő beaneket tudjuk regisztrálni.
- A @Configuration annotáció segítségével konfigurációs osztályokat hozhatunk létre, de az még nem akitválja a keretrendszer **Auto Configuration** lehetőségét.
- Van egy további módszer is, a @EnableAutoConfiguration, viszont ez nem szerepel a listában.
- Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/using.html#using.auto-configuration

6. feladat 0/1 pont

Milyen parancssori argumentummal megadásával tudjuk rábírni a Spring-et, hogy az általa használt automata konfigurációk listáját kiírja a konzolra induláskor?

Válaszok

A helyes válasz:

--debug

debug

--trace

Magyarázat

A --debug paraméterrel bekapcsoljuk a DEBUG alapú loggolást, ezáltal a Spring Boot-on belüli AutoConfiguration osztályok listája megjelenik a kimeneten. Hasznos, hogyha nem tudjuk, hogy milyen beanek miért jönnek létre.

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#using.auto-configuration.replacing

7. feladat 0/1 pont

Mr. Hammond bár nem sajnálta semmire se a pénzt, viszont a fejlesztés első hetében az egyik fejlesztő szabadságra ment és "félmunkát" végzett, a tárolóba egy fékész kódot töltött fel. Töltsd le a következő állományt és bírd működésre az alkalmazást, majd válaszolj a következő kérdésekre!

Milyen függőségeket kell hozzáadni a projekthez, hogy sikeresen forduljon?

A 7.-10. feladatok megoldásához a letöltött állományt használd!

Válasz

- ○ org.springframework.boot:spring-boot-extension
- ○ org.springframework:spring-context
- ● org.springframework.boot:spring-boot-starter
  Ez a válasz helyes, de nem jelölted meg.
- ○ org.springframework:spring-webmvc

Magyarázat

Az alkalmazás elindításához a spring-boot-starter függőséget kell behúznunk, bármelyik másikat húzzuk be nem fog elindulni az elvárt módon. A org.springframework.boot:spring-boot-extension pedig nem is létezik.

8. feladat 0/1 pont

Milyen annotációt hagyott le a kolléga az **InGenConfiguration** osztályban lévő metódusról?

Válaszok

A helyes válasz:

@Bean

org.springframework.context.annotation.Bean

@org.springframework.context.annotation.Bean

Bean

Magyarázat

A **@Bean** annotáció segítségével tudjuk elérni, hogy a metódus regisztrálja be a létrehozott új példányt az alkalmazás kontextusába.

9. feladat 0/1 pont

Milyen Maven céllal (goal) lehet az alkalmazást elindítani? (mvn nélkül)

Válasz

A helyes válasz:

spring-boot:run

Magyarázat

A megadott Maven céllal tudjuk elérni, hogy az alkalmazás sikeresen elinduljon.

10. feladat 0/2 pont

Mi a jelenlegi InGen firmware verziója? (a konzolon a következőként kezdődik **Current InGen firmware version:** és a kettőspont utáni értéket keressük)

Válasz

A helyes válasz:

1990-11-20-0.12.2-alpha

Magyarázat

Miután sikeres megoldottunk minden problémát és elindítottuk az alkalmazást akkor a következő értéket kell látnunk.

Megjelenés

Világos