.NET MICROSERVICES

1. forduló

A kategória támogatója: DXC Technology

Ismertető a feladatlaphoz

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.

Több válaszlehetőség esetén a helytelen válasz megjelölése mínusz pontot ér.

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.

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 ChatGPT használata nem tiltott, de az arra való hivatkozással észrevételt NEM fogadunk el!

A feltűnően rövid idő alatt megoldott feladatlapok kizárást vonnak maguk után, bármilyen más gyanús esetben fenntartjuk a jogot a forduló érvénytelenítésére!

Jó versenyzést kívánunk!

1. feladat 3 pont

Mikor használjunk sealed classokat?

Válaszok

- [ ] Ha kifejezetten szeretnénk jelezni, hogy nem örökölhető le az osztályból.
- [ ] Sose, backward compatibility miatt maradt, használjunk abstract classokat helyette.
- [ ] Minden lehetséges esetben érdemes, segít a compilernek optimalizálni a kódot.
- [ ] A fentiek közül egyik sem.

2. feladat 3 pont

Milyen problémára kínál megoldást a Factory Pattern a szoftvertervezésben?

Válasz

- [ ] A Factory Pattern olyan struktúrát biztosít, amely lehetővé teszi a nagy adathalmazok gyors elemzését és vizualizációját.
- [ ] A Factory Pattern alkalmazható, ha egy osztály rendelkezik több konstruktorral, és egy központi helyen történő létrehozást és inicializációt igényel.
- [ ] A Factory Pattern lehetővé teszi a kódbázis egyszerűsítését a magasabb teljesítmény elérése érdekében.
- [ ] A Factory Pattern a hálózati kommunikáció hatékonyságának javítását célozza meg a kliens-szerver alkalmazásokban.

3. feladat 1 pont

Microservice alkalmazás fejlesztés során mit nevezünk "DTO"-nak?

Válasz

- A DTO egy debugging eszköz, amely segíti a microservice futás idejű belső állapotának vizsgálatát.
- A DTO egy tervezési minta, ami kikényszeríti a szigorú szeparációt a microservice közt, ezzel megakadályozva az adat szivárgást köztük.
- A DTO egy caching megoldás, amely tárolja a gyakran használt adatokat és ezzel segít optimalizálni a microservice teljesítményét.
- A DTO egy egyszerű objektum, amely egységbe zárja az adatot és segíti a hatékony kommunikációt a microservicek közt.

4. feladat 1 pont

Az alábbi web API egy microservice CRUD műveleteit definiálja:

CRUD	HTTP Method	Relative path
CREATE	POST	/api/createOrders/{id}
READ	GET	/api/getOrders
UPDATE	PUT	/api/setOrders/{id}
DELETE	DELETE	/api/deleteOrders/{id}

A fenti egy RESTful API?

Válasz

- Igen
- Nem

5. feladat 3 pont

Az alábbi kód egy aszinkron üzenetküldést szimulál.

```
namespace ConsoleApp1
{
    public class Program
    {
        public class Person
        {
            public string Name { get; }

            public Person(string name)
            {
                Name = name;
            }

            public async Task SendMessage()
            {
                Random rnd = new Random();
                int sendDelay = rnd.Next(500, 2001);
                await Task.Delay(sendDelay);
                Console.WriteLine($"{Name} elküldött egy üzenetet.");

                int receiveDelay = rnd.Next(500, 2001);
                await Task.Delay(receiveDelay);
                Console.WriteLine($"{Name} megkapott egy üzenetet.");
            }
        }

        static async Task Main(string[] args)
        {
            var pepe = new Person("Pepe");
            var adam = new Person("Adam");

            Task pepeMessageTask = pepe.SendMessage();
            Task adamMessageTask = adam.SendMessage();

            await Task.WhenAll(pepeMessageTask, adamMessageTask);

            Console.WriteLine("Üzenetküldés befejezve.");
        }
    }
}
```

Melyek lehetnek a fenti kód kimenetei?

Válaszok

- [ ] "Pepe elküldött egy üzenetet."

  "Adam elküldött egy üzenetet."

  "Pepe megkapott egy üzenetet."

  "Adam megkapott egy üzenetet."

"Üzenetküldés befejezve."

- [ ] "Adam elküldött egy üzenetet."

  "Pepe elküldött egy üzenetet."

  "Adam megkapott egy üzenetet."

  "Pepe megkapott egy üzenetet."

  "Üzenetküldés befejezve."

- [ ] "Adam elküldött egy üzenetet."

  "Adam elküldött egy üzenetet."

  "Pepe megkapott egy üzenetet."

  "Pepe megkapott egy üzenetet."

  "Üzenetküldés befejezve."

- [ ] Egyik sem lehet a három közül.

Megoldások beküldése