Szerver oldali .NET fejlesztés

Beadandó feladat

1. Készítsen alkalmazást, amely képes kezelni, hogy melyik oktató melyik félévben melyik tantárgyat oktatta, és melyik tantárgyat melyik hallgatók vették fel. Az **oktatóknak** van Neptun kódja, neve, email címe, beosztása. A **beosztásoknak** fix értékkészlete van: docens, adjunktus, mesteroktató, ügyvivő szakértő, tanársegéd, egyéb. A **tantárgyaknak** van neve, kódja, kreditértéke (egész szám), felelős tanszéke (pl: VIRT, RSZT, Matematika, stb.). A **féléveknek** van neve (pl: 2022/23/1), kezdő dátuma és végdátuma. A **hallgatóknak** van Neptun kódja, neve, email címe és szakja. A **szakoknak** fix értékkészlete van: Mérnökinformatikus Msc, Programtervező informatikus Msc, Mérnökinformatikus Bsc, Programtervező informatikus Bsc, Gazdaságinformatikus Bsc.
2. Egy tantárgyat oktathat több oktató egy félévben; egy félévben lehet több tantárgy is; egy hallgató felvehet több tantárgyat is egy félévben; egy tantárgyon több hallgató is lehet.
3. Az 1. és 2. pont alapján készítse el entitásokat és a közöttük lévő függőségeket is az alkalmazásban. Készítse el az adatbázist, ehhez használja az Entity Framework Core 6 (továbbiakban EF) migrációját. A továbbiakban minden adatbázis művelethez használja az EF szolgáltatásait.
4. Az alkalmazásban lehessen listázni az összes tantárgyat, oktatót, hallgatót, félévet külön-külön végpontokon.
5. Lehessen lekérni egy oktató egy félévben oktatott összes tantárgyát.
6. Lehessen lekérni egy hallgató egy félévben felvett összes tantárgyát.
7. Lehessen új oktatót, tantárgyat, félévet és hallgatót felvinni a rendszerbe, ezeket összerendelni és a meglévőket lehessen módosítani, illetve logikailag törölni. Készítse el továbbá a szükséges végpontokat ahhoz, hogy az adatok egymással összerendelhetőek legyenek.
8. Állítson be global query filtert, hogy a félévek, oktatók, hallgatók és tantárgyak közül csak a nem töröltek jöjjenek le az adatbázisból. Az összes listázást végző végpontnál lehessen megadni, hogy a töröltek is legyenek a listában vagy ne.
9. Egészítse ki a tantárgyak adatait órarendi információval. Ez legyen szabad szöveges mező, melynek alapértelmezett értéke „ismeretlen” legyen. A tantárgyakat létrehozó és módosító végpontokat egészítse ki úgy, hogy ezt az információt meg lehessen adni. Ennek a megadása legyen opcionális, ha nem ad mega felhasználó semmilyen értéket, akkor a mező értéke az előbbiekben leírt alapértelmezett értéket kapja meg.
10. Lehessen lekérni az oktató által adott félévben oktatott összes hallgató listáját. A listában legyen benne a hallgató neve, Neptun kódja és hogy az oktató melyik tantárgyán oktatja őt. A lista a tantárgykódok és a hallgatók neve alapján legyen növekvő sorrendbe rendezve. A listában legyen benne minden adat, beleértve a törölteket is.
11. Lehessen lekérni, egy listát, amely tartalmazza, hogy megadott félévben a megadott oktató összesen mennyi kreditértékű tantárgyat oktatott és összesen hány hallgató volt ezeken a tantárgyakon (egy hallgató csak egyszer legyen összeszámolva, hiába oktatta több tárgyon is az oktató).
12. Készítse el a Repository és a UnitOfWork réteget.
13. Készítsen UnitOfWorköt a tantárgyakhoz, a UnitOfWorkön keresztül legyen lehetőség lekérni tantárgyakat időintervallum alapján. Ezt a félévek kezdő és végdátuma alapján határozza meg olyan módon, hogy ha a megadott intervallum belelóg bármennyire is a félévbe, akkor annak a félévnek a tantárgyai már beleszámítanak a keresésbe.

Szerver oldali .NET fejlesztés

Beadandó feladat 2022

14. Egészítse ki a rendszert az IdentityUser használatával úgy, hogy lehessen felhasználóknak regisztrálni, bejelentkezni és kijelentkezni. A felhasználónak legyen születési dátuma, neve, Neptun kódja és oktató esetén tanszéke (ez a mező nem oktató esetén „ismeretlen” értékű legyen).
15. Alakítsa át az entitásokat úgy, hogy minden oktató és hallgató felhasználó legyen.
16. Használja az authentikációs és az authorizációs middlewaret. Az authentikáció JWT tokent állítson elő és a program azt használja a továbbiakban.
17. Hozzon létre szerepköröket a rendszerben az IdentityRole használatával és ezeket rendelje a felhasználókhoz. A szerepkörök legyenek Admin, Teacher és Student. A felhasználó szerepkörét a claimjében tároljuk.
18. Készítsen egy middlewaret, amely hozzárendel egy azonosítót (Guid) a kéréshez, majd a választ a következő formába alakítja át:
    a. {
        i. „statusCode”: 200/404/500 stb.
        ii. „content”: maga a visszaküldendő adat
        iii. „identity”: a kérés beérkezésekor generált Guid
    b. }
19. Készítsen legalább 3 felhasználót a rendszerben, akik közül az egyik Teacher a másik Admin a harmadik Student szerepkörrel rendelkezzen. Amennyiben a program indulásakor valamelyik szerepkörhöz nincs legalább 1 felhasználó az adatbázisban, akkor az induláskor automatikusan jöjjön létre felhasználó arra a szerepkörre.
20. Készítsen végpontot, amelyen keresztül regisztrálható egy új felhasználó, ekkor automatikusan Student legyen a szerepköre.
21. Készítsen végpontot, amelyen keresztül egy felhasználó szerepköre megváltoztatható.
22. Az Authorize illetve az AllowAnonymus attribútumok használatával készítsen authorizációt a végpontokhoz a következő módon:
    a. A regisztrációt, a bejelentkezés és a tantárgyak listázását bárki elérhesse.
    b. Egyéb lekéréseket (GET végpontok) csak regisztrált felhasználók érhessék el.
    c. Az adatfelvitel, módosítás, törlés, összerendelés végpontokat csak Admin szerepkörrel lehessen elérni.
    d. Készítsen egy új végpontot, amely a 4 vagy annál nagyobb kreditértékű tantárgyakat listázza, az egyes tárgyaknál az azt oktató oktatókkal együtt. A végpontot a Student vagy Teacher szerepkörű felhasználók hívhassák meg.
    e. Készítsen policy-t, amely kiszűri, hogy csak az aktív (nem törölt) hallgatók és oktatók kérhessék le a 4 kredit vagy annál magasabb kreditértékű tantárgyakat az előző végponton.
    f. Készítsen middlewaret, amely hallgatói lekérés esetén a tantárgyak közül csak azoknak az adatait engedi visszaküldeni, amelyeket a hallgató hallgat.
23. Request logging témakörben tanultak alapján írassa ki a konzolra/fájlba a beérkező kéréseket, de csak azokat, amelyek megkövetelik az authorizációt. A logban legalább a következő adatok:
    a. A kérés http metódusa
    b. A kérés végponjának neve
    c. A kérés törzse JSON vagy XML formátumban
    d. A kérést indító felhasználó azonosítója
    e. A kérést indító felhasználó neve

Szerver oldali .NET fejlesztés

Beadandó feladat 2022

24. Memóriában cachelje a tantárgyakat. Alakítsa át úgy a service-eket, hogy a tantárgyak esetében a cache-t használják, ügyeljen arra, hogy a cache konzisztens legyen minden esetben az adatbázissal.
25. Használja és konfigurálja a Swaggert az alkalmazásban az API dokumentáció generálására.
26. Használja a modell validációs attribútumokat a modelljeinek property-jein. A property-ket próbálja értelemszerűen ellátni ezekkel az attribútumokkal.
27. Készítsen custom modell validációs attribútumot a Neptun kódhoz. A Neptun kód pontosan 6 karakterből áll, csak számokat vagy betűket tartalmazhat, és nem kezdődhet számmal.
28. **Opcionális**: Szervezze ki az adatbázis kapcsolódás konfigurációját az appsettings.json fájlba. A konfiguráció használatához az alkalmazásban használja az Options pattern megvalósítását.

2-es szint: az első 13 pont maradéktalan megvalósítása, vagy a kimaradó feladatpontok helyett a többi feladatpontból azokkal ekvivalens pontok megoldása.