Kapcsolattartók dokumentáció

Tartalomjegyzék:

Felhasználói végpontok: 1
Adatbázisok: 2
Company tábla: 2
Contact tábla: 2
Diagrammok 3
Egyedkapcsolat diagramm 3
Osztálydiagramm 3
Use Case Diagramm 4
Tesztelés: 4

Felhasználói végpontok:

(Minden felhasználói végpont teszteléséhez a Postman alkalmazás lett használva)

1. **Összes kapcsolattartó listázása:**

   Ezzel a funkcióval a user képes az összes adatbázisbeli kapcsolattartót lekérdezni, ezt a

   http://localhost:8080/contact/all elérési úttal tudja megjeleníteni.

2. **Id alapján megadott kapcsolattartó lekérdezése:**

   A felhasználó továbbá el tudja érni bármelyik kapcsolattartót az id-ja alapján, majd meg is tudja nézni a keresett személy összes adatát. Az utóbbi eredményt a http://localhost:8080/contact/id oldalon tudja elérni, ahol az id egy szám, ami kapcsolattartótól függő.

3. **Új kapcsolattartó felvétele:**

A felhasználó fel is tud venni új kapcsolattartót a http://localhost:8080/contact/add oldalon, itt kötelezően meg kell adnia a kapcsolattartó teljes nevét, e-mailcímét, státuszát és a cégének a nevét, illetve megadhatja annak telefonszámát és kommentet is fűzhet hozzá.

4. **Kapcsolattartó törlése (DELETED státuszra változtatás):**

   Ezzel a funkcióval a user egy általa választatott kapcsolattartónak a státuszát tudja ACTIVE-ról DELETED-re változtatni, ezzel az adatbázisból még nem törlődik a személy viszont, a fő oldalon már nem fog megjelenni. Ezt a felhasználó a http://localhost:8080/contact/deactivate/id oldalon tudja elérni, ahol az id a megadott kapcsolattartó id-ja.

5. **Kapcsolattartó végleges törlése:**

   A felhasználó végleges is törölni tudja az adatbázisból a kiválasztott személyt a http://localhost:8080/contact/delete/id eléréssel, ahol az id a kapcsolattartó id-ja.

6. **Kapcsolattartó frissítése:**

   A usernek lehetősége van egy kapcsolattartó bármelyik adatának megváltoztatására, itt először betöltődik a kiválasztott kapcsolattartó és annak minden adata, és ennek a személynek lehet majd bármelyik adatát átírni, majd a megfelelő mezők frissülni fognak az adatbázisban is. A funkciót ezen az elérési útin lehet használni, http://localhost:8080/contact/update/id, ahol az id a kiválasztott kapcsolattartó id-ja.

Adatbázisok:

Company tábla:

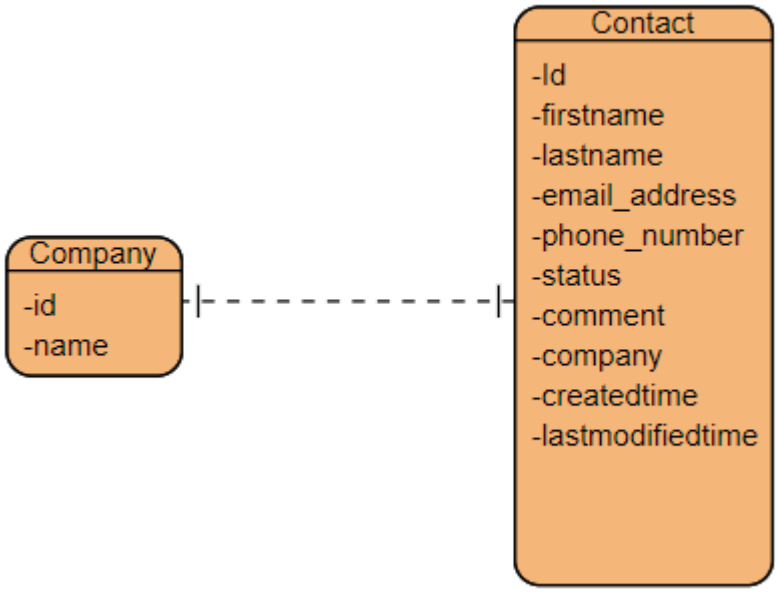
Név	Típus	Kikötés
Id	Long	Nem lehet üres
Name	String	Nem lehet üres

Contact tábla:

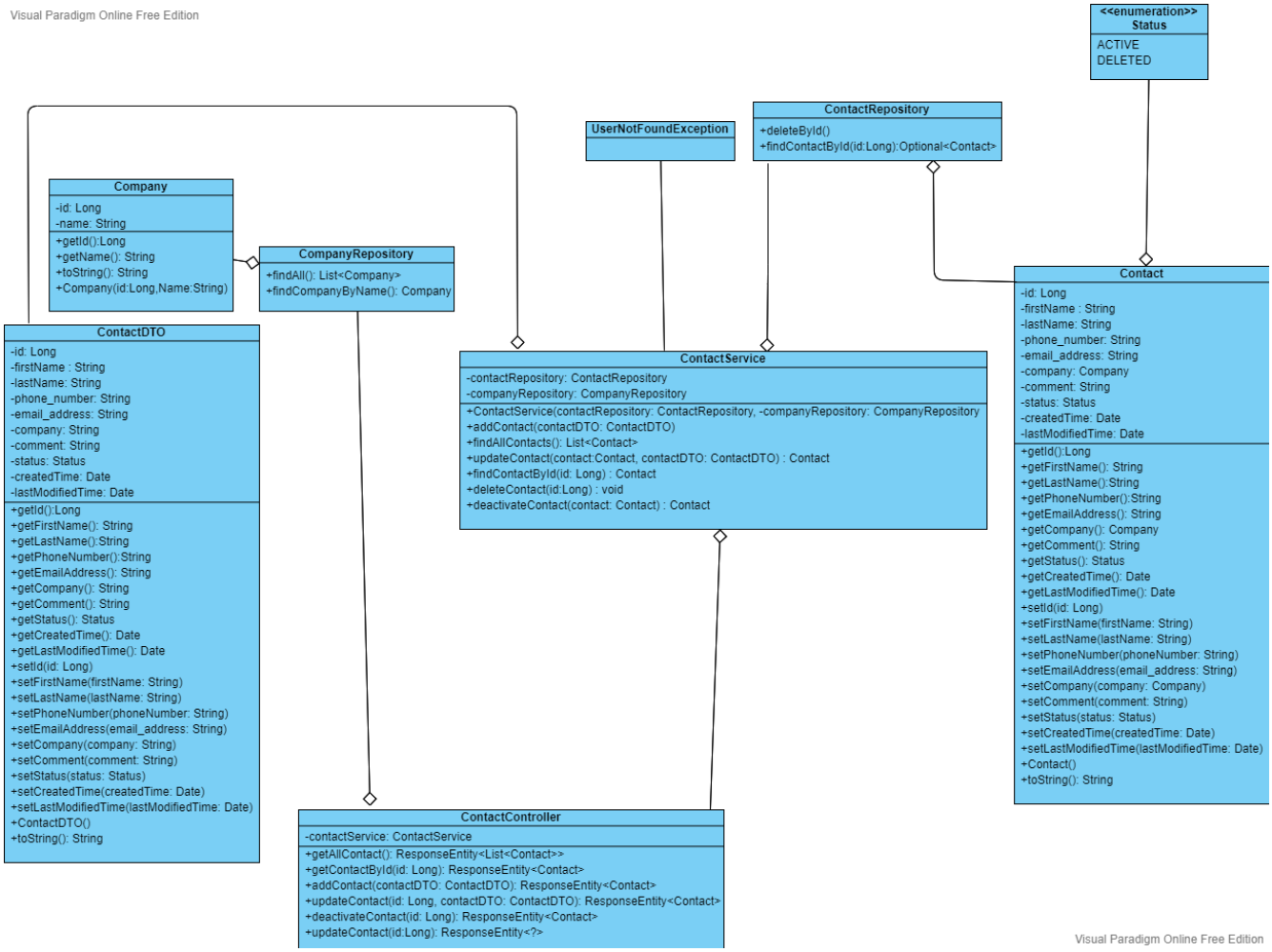
Név	Típus	Kikötés
Id	Long	Nem lehet üres
firstName	String	Nem lehet üres
lastName	String	Nem lehet üres
phoneNumber	String	Opcionális, E-164 szabványnak megfelelő
emaillAddress	String	Nem lehet üres, E-Mail formátum
company	Hozzárendelés	Nem lehet üres, Kiválasztható egy darab cég az adatbázisból
status	Enumeration	Nem lehet üres, ACTIVE és DELETED a választható
comment	String	
createdTime	Date	
lastModifiedTime	Date	

Diagrammok

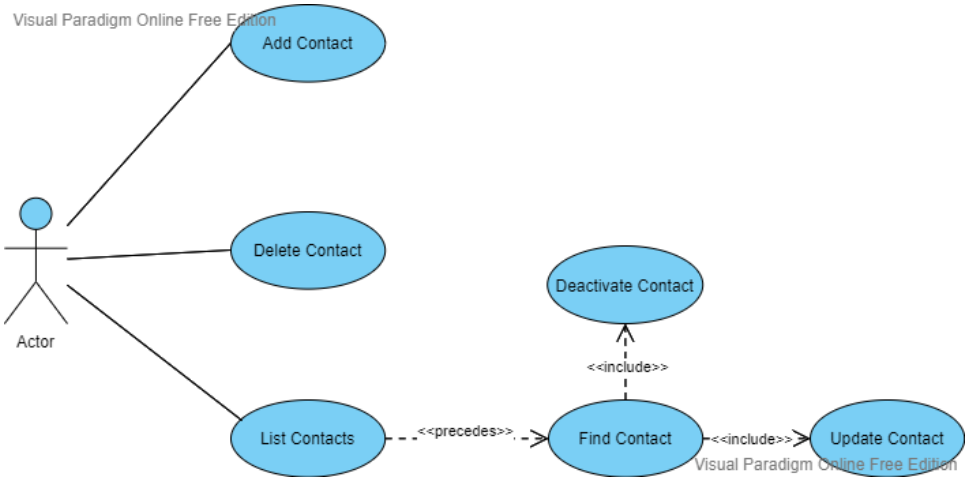
Egyedkapcsolat diagramm



Osztálydiagramm



Use Case Diagramm



Tesztelés:

A következő rendszer komponensek lettek tesztelve unit tesztekkel:

I. **CompanyRepository tesztelése:**
   1. **findAllTest():** Ebben a tesztben az előre megírt findAll metódus lett tesztelve először még üres adatbázissal.

2. **findAllTestAfterAdd():** Ebben a tesztben az üres adatbázist bővítettük 3 új Companyvel és utána a findAll-lal lekérdeztük az összeset, így kiderült, hogy helyesen, 3 darab új elemmel bővült az adatbázis.
3. **findCompanyByNameTest():** Ebben a példadában az üres adatbázishoz felvettünk 3 új elemet, majd a Company neve szerint kikerestünk egy Companyt, ehhez a findComepanyByName függvény lett használva.

II. ContactRepository tesztelése:

1. **deleteByIdTest():** A teszt során az in-memory adatbázisba felvételre került egy kapcsolattartó, akit a deleteById(Long) metódussal töröltünk is az adatbázisból**.**
2. **findContactbyIdTest():** A tesztben hozzáadtunk egy új kapcsolattartót az adatbázishoz, majd a findContactById(Long)-val megkerestük azt, az Id-ja segítségével.

III. ContactService tesztelése:

1. **addContactTest():** A tesztelés során a mockolt ContactRepository-t használtuk arra, hogy ellenőrizzük a ContactService addContact(ConcatcDTO) metódusát, ahol a végén össze lett hasonlítva az adatbázisba felvett és a helyben létrehozott kapcsolattartó.
2. **findAllContactsTest():** A tesztben a mockolt ContactRepository lett használva a ContactService tesztelésére, azon belül is a findAll() függvény lett tesztelve.
3. **updateContactTest():** Ebben a tesz esetben egy előre létrehozott kapcsolattartót vettünk fel az adatbázisba, majd firstname-jét és lastname-jét megváltoztattuk a ContactService updateContact függvényével, és elmentettük a változtatásokat az adatbázisba.
4. **deleteContactTest():** A teszt során a mockolt ContactRepositoryval lett ellenőritve a ContactService deleteContact(Long) metódusa, amivel egy kapcsolattartót tudtunk törölni az adatbázisból.
5. **deactivateContactTest():** Ebben a teszt esetben a mockolt ContactRepository segítéségel lett ellenőrizve a ContactService osztály, aminek itt a deactivateContact metódusa lett tesztelve az által, hogy egy kikeresett kapcsolattartónak a státuszát ACTIVE-ról DELETED-re változtattuk.