

Webalkalmazás fejlesztés (Java)

2020 őszi félév – feladatkiírás

Opcionális!

Feltöltési határidő GitHub: 2020.12.08. 10:00

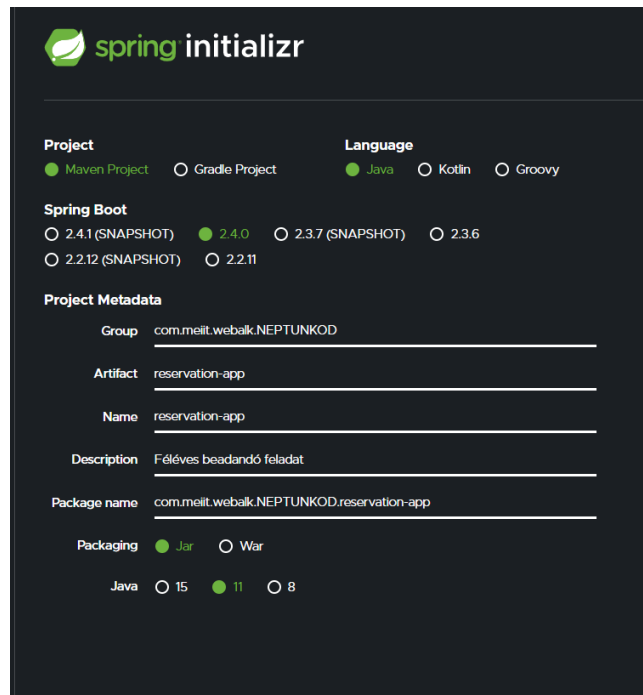
A feladat prezentálása és megvédése 2020.12.08. és 12.10. online formában lehetséges a kiírt előadás és gyakorlati időpontokban.

Hotelfoglaló alkalmazás

Valósítson meg egy **Hotelfoglaló webes alkalmazást** a korábban elkészített Java feladat felhasználásával.

Készítsen a következő weboldal segítségével egy Spring-boot alkalmazást.
<https://start.spring.io/>

Az oldalon a következőket kell megadni:



The screenshot shows the Spring Initializr web application configuration page. The page is dark-themed with green accents. It includes sections for Project, Language, Spring Boot, Project Metadata, and Packaging. The Project section has radio buttons for Maven Project (selected) and Gradle Project. The Language section has radio buttons for Java (selected), Kotlin, and Groovy. The Spring Boot section has radio buttons for 2.4.1 (SNAPSHOT), 2.4.0 (selected), 2.3.7 (SNAPSHOT), 2.3.6, 2.2.12 (SNAPSHOT), and 2.2.11. The Project Metadata section includes fields for Group (com.meit.webalk.NEPTUNKOD), Artifact (reservation-app), Name (reservation-app), Description (Féléves beadandó feladat), and Package name (com.meit.webalk.NEPTUNKOD.reservation-app). The Packaging section has radio buttons for Jar (selected) and War. At the bottom, there are radio buttons for Java 15, 11 (selected), and 8.

Add dependencies menü alatt a következőket adja hozzá:

- **Lombok** -> egyszerűsíti a kódírást gyorsabban lehet vele haladni (Opcionális)
- **Spring Web**
- **Thymeleaf** -> a megjelenítés egyszerűbben megoldható HTML segítségével.
- **Spring Security** -> login funkcióhoz van rá szükség (amíg a login nincs implementálva commentezze ki a dependecyt, hogy a feladat nagyrészt el tudja addig végezni).
- **Spring Data JPA**
- **H2 database** -> memóriában tárolt db-t lehet vele létrehozni (optional).

- **MySQL Driver**
- **Validation** -> bemeneti értékek validását könnyíti meg

GENERÁLJA LE A PROJEKTET

Az így letöltött mappát csomagolja ki, majd importálja be a fejlesztő környezetbe, mint project-et.

A pom.xml-ben adja hozzá a következő dependencyket:

```
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger2</artifactId>
  <version>2.9.2</version>
</dependency>
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger-ui</artifactId>
  <version>2.9.2</version>
</dependency>
```

A rest endpoint-okat ezzel fogjuk tesztelni.

Hozza létre a következő maven modulokat, mindegyik tartalmazzon saját pom.xml file-t!

reservation-app-controller

reservation-app-domain

reservation-app-service

A generált java classt, amely tartalmazza a main metódusokat helyezze át a reservation-app-controller/src/main/java/com.meiit.webalk.NEPTUNKOD nevű mappájába.

Az application main method-ját tartalmazó class-t egészítse ki az annotációval.

```
@EnableSwagger2
```

Megvalósítás

reservation-app-controller

Felépítés

A reservation-app-controller/src/main/java/com.meiit.webalk.NEPTUNKOD package csak a korábban említett main osztályt tartalmazza, valamint egy Önök által létrehozott package-et aminek a neve **web** lesz.

A reservation-app-controller/src/main/java/com.meiit.webalk.NEPTUNKOD.web package-ben a következő struktúra megvalósítása a kötelező:

com.meiit.webalk.NEPTUNKOD.web.config: ezen belül SecurityConfig.java osztály a spring security beállításához.

com.meiit.webalk.NEPTUNKOD.web.controllers: ezen a package-n belül minden funkcióhoz hozzanak létre külön package-t, amit a funkciónak megfelelően neveznek el. Minden package-n belül legalább egy az általa megvalósított viselkedést tükröző névvel létrehozott ValamiController.java osztályban implementálják a controllereket. pl: LoginController, MainPageViewController, DetailsFormController stb.

extra: A megjelenítendő adatok nem mindig egyeznek a db-ből kinyertekkel, valamint nem biztonságos közvetlenül a db-ben használt domain osztályokat használni. Ahol jónak látja használhat DTO osztályokat ezt külön package-be helyezze el és használjon transformer osztályokat az entity -> dto valamint dto -> entity átalakításra.

reservation-app-controller/src/main/resources/application.properties file tartalmazza a db configot.

Valamint a következő frontendre használatos beállításokat:

```
spring.thymeleaf.check-template=true
spring.thymeleaf.enabled=true
spring.thymeleaf.encoding=UTF-8
spring.thymeleaf.prefix=classpath:/templates/WEB-INF/
```

reservation-app-controller/src/main/resources/templates/WEB-INF mappában hozza létre az alkalmazás oldalait reprezentáló .html file-kat.

Ezeknek a későbbiekben látható mintának kell megfelelnie. Használjanak CSS Bootstrap-et (úgy könnyebb és gyorsabb).

reservation-app-domain

Felépítés

reservation-app-domain/src/main/java/com.meiit.webalk.NEPTUNKOD-mappában a java feladat alatt implementált domain osztályokat tároljuk. Ezekben használják a JPA-nál tanult annotációkat, úgy hogy azok egy adatbázist reprezentáljanak.

reservation-app-domain/src/main/java/com.meiit.webalk.NEPTUNKOD.repositories mappában található osztályokhoz tartozó spring data repository interfacek megvalósítása a feladat. Szükség esetén az interfacek kiegészíthetők további lekérdezésekkel.

reservation-app-service

Felépítés

reservation-app-services/src/main/java/com.meiit.webalk.NEPTUNKOD.services mappában a funkcióként elnevezett mappában a szükséges ValamiService elnevezésű osztályokat kell implementálni. Minden Servicenek legyen saját Interface!

Ezekben a Service osztályokban implementálják a működéshez szükséges logikákat.

H2 adatbázis esetén:

reservation-app-services/src/main/java/com.meiit.webalk.NEPTUNKOD.testdata

Ezen a mappán belül hozzanak létre egy osztályt, amely legalább 1 test usert, valamint legalább 1 test hotelt szűr be az adatbázisba induláskor. Springben ahhoz, hogy induláskor meghívódjon egy adott függvény a következő megoldás kell:

```
public class TestDataGenerator implements CommandLineRunner
```

A CommandLineRunner interfaccal kötelező egy run metódus implementálása, ami az alkalmazás indulásakor mindig lefut.

```
@Override
public void run(String... args) throws Exception {
    createPerson();
    createTestHotels();
}
```

Ezzel feltölthető a H2 adatbázis. Más esetben adatokkal rendelkező adatbázis szükséges a bemutatáshoz.

Elvárt működés

Hozza létre az @RestController annotáció segítségével a következő RESTEndpoint-okat.

add-hotel -> Hotel hozzáadása minden adattal.

add-room -> szoba hozzáadása létező hotel adatokhoz.

extra: add-... -> emelet, szárny és egyéb hotellel kapcsolatos lehetőségek bővítése

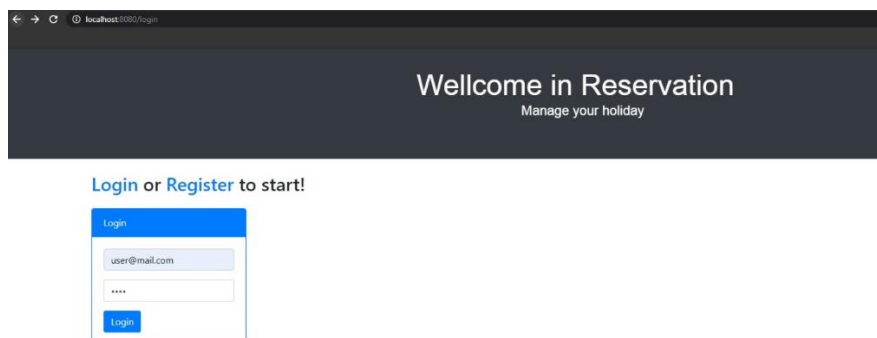
create-user -> user és person adatok hozzáadása

user-info -> meglévő user összes adatát adja vissza id alapján.

show-reserveations -> visszaadja az összes foglalást

Ezeket a <http://localhost:8080/swagger-ui.html> oldalon tudjuk ellenőrizni.

Megjelenés



A login funkció mellé a logout gombot is működőképesé kell tenni (egyszerű config a megoldás nem kell semmit implementálni hozzá).

Amennyiben még nincs foglalás a következő oldal jöjjön be login után:

[Reservation](#) [Home](#) [Hotels](#) [Language](#) Logout

Person details

Name

Dummy Dóme

Date of birth

1992-03-04

Account number

222333666

Currency

HUF

€

Balance

999.00

Save

A felhasználó ezen az oldalon változtathatja meg az adatait. Az egyszerűség kedvéért csak a név és az account változtatható.

Save gombra kattintva az adatok a db-be íródnak.

Amennyiben rendelkezik a felhasználó foglalással a details alatt a következő táblázat jelenik meg:

[Reservation](#) [Home](#) [Hotels](#) [Language](#) Logout

Person details

Name

Dummy Dóme

Date of birth

1992-03-04

Account number

222333666

Currency

HUF

€

Balance

899.00

Save

Books

#	Hotel	Reservation detail1	Reservation detail2
1	Hilton		

Remove

A listában jelenjen meg az összes foglalás minden adatával. Minden foglaláshoz tartozzon egy *remove* gomb, arra kattintva szűnjön meg a foglalás és törölje azt a db-ből is.

A navigációs fejléc Hotels menüpontjára kattintva jöjjön be egy Hotels táblázat, amiben az első oszlopban egy *book* gomb található, a többiben pedig a hotel adatai (név, cím, csillagok száma).

A book gombra kattintva navigáljunk tovább a reservation oldalra. Ahol egy fix táblázatban jelenjen meg a korábban is látott hotel adatai (most gomb nélkül).

Alatta felsorolásban jelenjen meg az összes szint a hozzá tartozó szárnnyal. Itt ismét soronként elhelyezünk egy gombot, amivel választhatunk az ajánlatok közül.

A gombra kattintva navigáljunk a szoba választó oldalra.

Korábbihoz hasonlóan fix táblázatban jelenítsük meg az eddig ismert adatokat a hotelről, a szintről stb.

Alatta legyen választható 1 legördülő menüből a szoba. A legördülő menü megjelenített értéke legyen: Ár: xxxxx , szoba szám: x, ágyak száma: x (amennyiben a szoba erkélyes az értékben jelenjen meg az is).

Továbbá legyen itt 2 gomb, a create reservation és cancel.

Create reservation az adatbázisban létrehozza a foglalást az adatokkal, majd redirect-eli a usert a main page-re.

Cancel visszairányít az előző oldalra.

Style: A beszúrt képekhez hasonló stílusban valósítsa meg az alkalmazás összes oldalát.

A színek változhatnak, a betűméreteket tetszés szerint döntsék el a lényeg, hogy olvasható legyen az oldal.

Dr. Bednarik László
tárgyjegyző