Информациони системи 1/Пројекат

< Информациони системи 1

Пројекат на предмету носи 20 бодова и брани се у прва два испитна рока. Може да се ради само једном и бодови са испита могу да се пренесу у наредни рок уколико се тада брани пројекат. Обухвата цело градиво рађено на вежбама, али такође може да дође градиво које је поменуто али не и рађено на вежбама, па је понекад потребно сналазити се по ресурсима на интернету. На одбрани пројекта могу да дођу модификације, али се то од почетка 2022. године није још увек десило.

Пројекат се мења сваке године, али обично се ради о централном серверу (*Web Application* пројекат као на другој лабораторијској вежби) који комуницира са подсистемима (*Enterprise Application Client* пројекат као на првој лабораторијској вежби) преко JMS, само подсистеми имају приступ својим базама података (користе JPA) и постоји клијентска апликација (клијентски део се обично не ради на вежбама) која са централним сервером комуницира преко HTTP протокола (позивајући његов REST API). Водич испод служи како би напоменуо неке честе грешке у изради пројекта.

Препоручује се да пре израде пројекта урадите лабораторијске вежбе и прочитате водиче за JMS, JPA и REST. Уколико сте радили лабораторијске вежбе, за пројекат је обично потребно од 3 до 7 дана.

Клијент

Клијентски део пројекта треба да комуницира са REST сервером преко HTTP протокола, али се коришћење HTTP клијената обично не ради на вежбама. На консултацијама за пројекат 2022. године (https://studentetfbgacrs-my.sharepoint.com/:v:/g/personal/sa190595d_student_etf_bg_ac_rs/Eb6uDGtj-wtDpgpUrWBFeXUB7c2IEK_hNTbrXKneuJodDA) (11:12) Стефан Тубић је показао како се прави *Java with Ant → Java Application* пројекат и у њега убацује *JAX-RS 2.0* библиотека, а примери употребе су дати на презентацији на страници предмета, (http://si3is1.etf.rs/Nastava/Vezbe/3%20-%20JPA%20i%20REST%20API/2%20-%20RESTful%20web%20servisi.pdf) последња три слајда. На консултацијама није покренуо ниједан HTTP захтев користећи код са презентација, и зато су се студентима дешавале грешке када би покушавали да раде на тај начин. Није познато да ли је иједан предати пројекат досад користио ту методу.

Један приступ писању HTTP клијентске апликације јесте коришћење `HttpURLConnection` (https://docs.oracle.com/javase/8/docs/api/java/net/HttpURLConnection.html) из Јава стандардне библиотеке, али она нуди интерфејс који може бити превише ниског нивоа за потребе пројекта (као што је и нормално у стандардним библиотекама програмских језика). Због тога се препоручује коришћење библиотека попут *Retrofit*, (https://square.github.io/retrofit/) које неке детаље HTTP комуникације сакривају од корисника, дајући једноставнији интерфејс.

Уколико користите *Retrofit*

- Уместо *Ant* пројекта, препоручује се *Maven* пројекат у коме за увожење библиотеке можете да додате следеће у `pom.xml` фајл, унутар `<dependencies>`:

```
<dependencies>
    <dependency>
        <groupId>com.squareup.retrofit2</groupId>
        <artifactId>retrofit</artifactId>
        <version>2.8.1</version>
    </dependency>
    <dependency>
        <groupId>com.squareup.retrofit2</groupId>
        <artifactId>converter-gson</artifactId>
        <version>2.8.1</version>
    </dependency>
</dependencies>
```

  - Ако су изашле нове верзије можете да ажурирате на њих, а ако не користите *Gson* конвертер (за JSON) већ неки други, можете да замените са конвертером који користите.
- Препоручује се да на својим класама ресурса имате `@Produces(MediaType.APPLICATION_JSON)` анотацију како би враћали податке у JSON формату уместо XML. У овом случају могу да се десе грешке цикличне серијализације јер нема анотација које би спречиле то (постоји анотација `@JsonbTransient` која би требало да буде слична `@XmlTransient` анотацији за JSON, али понекад може да не ради) па у том случају пре враћања ресурса морате да прођете кроз његова инверзна поља и поставите их на `null`, или у потпуности избаците инверзна поља.
- Најбитнија документација се налази на страници пројекта, (https://square.github.io/retrofit/) а можете погледати и један од примера употребе са *GitHub* репозиторијума.

Честе грешке

- Уколико за пројекат додајете JMS комуникацију у REST пројекат, не заборавите да се JMS *Connection Factory*, *Topic* или *Queue* може убацити преко *Resource Injection* (користећи `@Resource` анотацију) само уколико је поље у које се убацују означено као нестатичко. Грешка која се појављује због овога гласи *Illegal use of static field on class that only supports instance-based injection*.
- Уколико морате да у једној методи пошаљете а затим чекате на одговор у тој истој методи и дешава вам се да се порука не пошаље пре него што се чека на захтев, проблем је у томе што се налазите у EJB (*Enterprise Java Bean*) контексту и покренута је трансакција па се тек на крају те трансакције пошаље тражена порука. Ово се може решити на два начина:
  - Са сваке класе ресурса уконити `@Stateless` анотацију. Ово има ефекат да се ваше методе више не извршавају у EJB контексту, па се трансакције не дешавају и поруке се шаљу одмах.
    - Уколико сте којим случајем користили `@Singleton` анотацију као начин реупотребе кода пређите са EJB Singleton (`javax.ejb.Singleton`) на CDI Singleton (`javax.inject.Singleton`) и убацујте га користећи `@Inject` уместо `@EJB` анотације у ресурсе.
  - (или) На сваки метод ресурса поставите `@TransactionAttribute(TransactionAttributeType.NOT_SUPPORTED)` анотацију. Ово има ефекат да те методе не буду обухваћене трансакцијама, па се проблем слично решава.
- Може да вам се деси грешка која пријављује да ентитетске класе можда нису означене са `@Entity`. То се обично дешава у случају да вам је `transaction-type` постављен на `RESOURCE_LOCAL` али немате све класе излистане у `persistence.xml` под `<class>` таговима.

- Уколико је `transaction-type` постављен на `RESOURCE_LOCAL` и добијате грешку `java.lang.NoClassDefFoundError: sun/security/ssl/HelloExtension` или `java.lang.NoSuchMethodError: sun.security.ssl.SSLSessionImpl`, у `persistence.xml` додајте:

```
<property name="eclipselink.jdbc.property.useSSL" value="false"/>
<property name="eclipselink.jdbc.property.requireSSL" value="false"/>
```

  - На лабораторијској вежби се ова грешка не дешава јер MySQL сервер покренут тамо не подржава SSL.
- Уколико вам се деси грешка са описом: *An attempt was made to traverse a relationship using indirection that had a null Session. This often occurs when an entity with an uninstantiated LAZY relationship is serialized and that relationship is traversed after serialization. To avoid this issue, instantiate the LAZY relationship prior to serialization.* могуће је да сте послали ентитетски објекат преко JMS у неки други адресни простор где тај објекат више није у истом JPA контексту, а нека поља су остала неиницијализована због `FetchType.LAZY` на тој асоцијацији, па је неуспешно покушана иницијализација тих поља док је серијализатор обилазио објекат како би серијализовао његова поља (дешава се без обзира на то да ли су поља означена са `@XmlTransient` или не). Ово можете решити тако што та поља поставите на `null` пре серијализације.
- Уколико добијате `org.omg.CORBA.COMM_FAILURE` грешке, најлакше решење је да рестартујете рачунар. Друго решење је да насилно погасите све *Java*, *NetBeans* и *GlassFish* процесе на које можете да наиђете. Ова грешка се обично дешава због тога што је више *GlassFish* инстанци покренуто.
- Уколико правите нове контексте и *JMS consumer* при сваком захтеву, требало би да позовете `close()` над њима, или да их поставите у *try-with-resources* (https://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html) како би се аутоматски затворили. Ово може да буде потенцијално решење за грешку изнад.

Напомене

- Када се од вас траже *dump* фајлови базе за предају, можете их наћи у *MySQL Workbench* под *Server → Data Export/Import*.
- Препоручује се да у пројектима подсистема (пројектима који нису *Web Application*) користите `transaction-type="RESOURCE_LOCAL"` јер постављање JTA да ради у тим пројектима може да буде тешко. У том случају није потребно стварање никаквих JDBC ресурса на *GlassFish* серверу, јер се креденцијали читају из `persistence.xml`, али је потребно ручно покретање и завршавање трансакција како би се успешно унеле промене у базу.
- UML дијаграме који се траже у пројекту можете да генеришете користећи неки алат попут *StarUML* са *Java* додатком инсталираним кроз *Tools → Extension Manager* (претражити *Java*). Користећи тај додатак можете генерисати дијаграм из кода опцијом *Tools → Java → Reverse Code*. Можда је потребно мало улепшати дијаграме након генерисања.

Покретање на одбрани

Пошто је ово први пут да студенти доносе своје пројекте на лабораторијске рачунаре, могу наићи на неколико грешки на које при изради лабораторијских вежби нису наилазили.

1. Направите базу података на *MySQL* серверу користећи опцију сличну оној за прављење *dump* фајлова. Алтернативно, направите ручно базу а затим садржај свог фајла извршите као

регуларну *SQL* скрипту која ће направити табеле и попунити их подацима.

2. У *Services* → *Servers* додати *GlassFish* који се налази на директоријуму `C:\Program1\glassfish5`, осим ако већ није додат са те путање. Покренути и додати неопходне ресурсе за базу података или JMS.
   - Уколико нисте користили `transaction-type="JTA"` ни у једном `persistence.xml`, нису вам потребни ресурси за базу података на *GlassFish* и уколико имате `<jta-data-source>` у било ком `persistence.xml` можете га уклонити.
3. При креирању/отварању пројеката промените *JDK* на *JDK 1.8* (подразумевано је *JDK 11*) на свим пројектима. U *Ant* пројектима ова опција се налази на *Properties* → *Libraries*, док се у *Maven* пројектима налази на *Compile*.
4. У свим *Ant* пројектима морају да се додају библиотеке којима *GlassFish* не може да приступи јер је *NetBeans* инсталиран у директоријуму чија путања садржи размак. Библиотеке са списка копирати у фолдер чија путања нема размак и у једном пројекту под *Properties* → *Libraries* изабрати опцију *Edit* над библиотекама *EclipseLink (JPA 2.1)* и *Java EE 8 API*, уклонити све JAR фајлове учитане од стране библиотеке и заменити их JAR фајловима ископираним на путању без размака.
   - Путање JAR фајлова тих библиотека су на следећих местима:
     - *EclipseLink (JPA 2.1)* — `C:\Program Files\NetBeans\netbeans\java\modules\ext\eclipselink`
     - *Java EE 8 API* — `C:\Program Files\NetBeans\netbeans\enterprise\modules\ext\javaee-api-8.0.jar`
   - Такође је потребно додати и конектор за *MySQL* са `C:\Program1\glassfish5\glassfish\lib\mysql-connector-java-8.0.20.jar`.
   - Идеално је спаковати све ове фајлове заједно са пројектом, повезане релативним путањама.
5. У свим `persistence.xml` подесити да се на базу повезује са налогом `admin`, чија је шифра `123`.
   - Или боље, у SQL скрипти за генерисање базе направити новог корисника и дати му све привилегије и онда њега користити у `persistence.xml`.

```
CREATE USER 'is1'@'localhost' IDENTIFIED BY 'sifra';
GRANT ALL PRIVILEGES ON * . * TO 'is1'@'localhost';
```

   - Уколико нисте користили `transaction-type="RESOURCE_LOCAL"` ни у једном `persistence.xml`, креденцијали за базу се читају са JDBC ресурса на *GlassFish* серверу и они из датотеке нису релевантни.
