Prof. Dr. Jörg Hettel

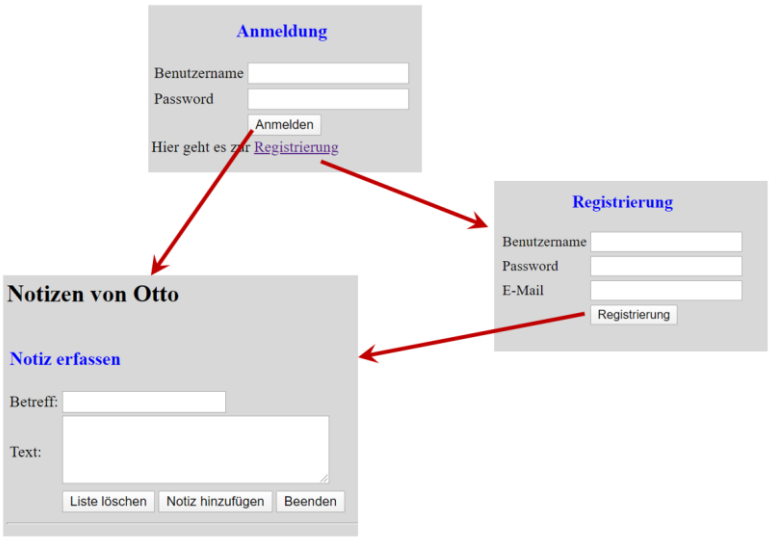
Datenbankprogrammierung
Arbeitspaket 4

Vorbemerkung

In diesem Aufgabenpaket geht es um den Datenbankzugriff innerhalb einer Java-Web-Applikation mit JPA. Hierzu wechseln Sie am besten in eine Entwicklungsumgebung, die die Entwicklung von Web-Anwendungen erlaubt, wie z.B. die *Eclipse for Java EE Developer* (siehe hierzu auch das Modul Web-Programmierung).

Auf der Web-Seite finden Sie eine kleine Web-Anwendung für die Verwaltung von Notizen. Laden Sie sich die Anwendung in Ihre Entwicklungsumgebung und machen Sie sich mit der Logik vertraut. Die Anwendung ist lauffähig, wobei eingegebene Daten nur transient gehalten werden.

Benutzer müssen sich zuerst registrieren. Nach erfolgreicher Registrierung wird der Benutzer gleich angemeldet und kann seine Notizen erfassen. Registrierte Benutzer können sich direkt einloggen.



Aufgabe 1 – Einrichten eines Schemas

Wenn Sie das vorherige Aufgabenblatt bearbeitet haben, haben Sie bereits die notwendige Datenbank und das benötigte Schema erzeugt. Hier nochmal die notwendigen SQL-Befehle:

```
DROP DATABASE  IF EXISTS usernotes;

CREATE DATABASE usernotes CHARACTER SET utf8mb4;

USE usernotes;

CREATE TABLE Users (id INT  PRIMARY KEY AUTO_INCREMENT,
                    username VARCHAR(50),
                    password VARCHAR(50),
                    email  VARCHAR(50) );
CREATE TABLE Notes (id INT  PRIMARY KEY AUTO_INCREMENT,
                    subject VARCHAR(100),
                    content VARCHAR(1024),
                    date  VARCHAR(100) ,
                    userId INT  );
```

Datenbankprogrammierung
Arbeitspaket 4

Aufgabe 2 – Einrichten einer DataSource

Vergewissern Sie sich, dass bei Tomcat auch eine entsprechende Datasource angelegt ist. In der **`server.xml`** Datei sollte bei den **`GlobalNamingResources`** folgender Eintrag zu finden sein:

```
<Resource auth="Container" driverClassName="com.mysql.cj.jdbc.Driver"
maxActive="10" maxIdle="3" maxWait="10000"
name="jdbc/Notes" type="javax.sql.DataSource"
url="jdbc:mysql://localhost/usernotes?useUnicode=true&amp;useLegacyDatetimeCode=false&amp;useJDBCCompliantTimezoneShift=true&amp;useLegacyDatetimeCode=false&amp;serverTimezone=UTC"
username="root"/>
```

Und in der **`context.xml`** Datei:

```
<ResourceLink name="jdbc/Notes" global="jdbc/Notes" type="javax.sql.DataSource" />
```

Vergewissern Sie sich auch, dass der entsprechende JDBC-Treiber bei Tomcat hinterlegt ist.

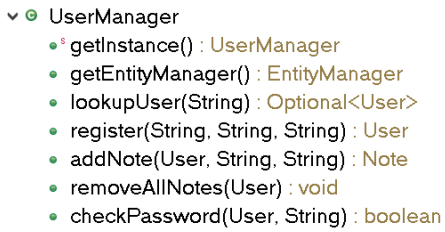
Aufgabe 3 – Persistence.xml

Um ein JPA-Framework nutzen zu können, muss analog zum JDBC-Treiber auch eine entsprechende JPA-Implementierung bereitgestellt werden. Als Implementierung soll *EclipseLink* verwendet werden, das nun ebenfalls ins lib-Verzeichnis der Anwendung kopiert werden muss:

Aufgabe 3 – Annotierung und Modifikation der Fachklassen

Damit die Persistenz der beiden Klassen **`User`** und **`Note`** vom JPA-Framework übernommen werden kann, müssen diese jetzt mit entsprechenden Annotierungen ausgezeichnet werden. Beachten Sie, dass zwischen **`User`** und **`Note`** eine unidirektionale 1:n Beziehung existiert.

In der Anwendung bietet es sich an, die Geschäftslogik (im Wesentlichen sind das die CRUD-Operationen) hinter einem Fassaden-Objekt zu verbergen:



Damit das JPA-Framework ordnungsgemäß funktioniert, muss noch die **`persistence.xml`** Datei zur Verfügung gestellt werden. Diese muss in das META-INF-Verzeichnis des **`src`**-Ordners gelegt werden. Dieses Verzeichnis muss in der Regel erst explizit erzeugt werden.

Da das JPA-Framework die bei Tomcat hinterlegt *DataSource* nutzen soll, sieht der Inhalt der Datei jetzt wir folgt aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence persistence_1_0.xsd"
version="1.0">

  <persistence-unit name="UserNotes" transaction-type="RESOURCE_LOCAL">

   <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
   <non-jta-data-source>java:/comp/env/jdbc/Notes</non-jta-data-source>

   <class>demo.model.Note</class>
   <class>demo.model.User</class>

   <properties>
      <!-- Uncomment to get log sql log output -->
      <!-- <property name="eclipselink.logging.level" value="OFF" /> -->
      <property name="eclipselink.logging.level.sql" value="FINE"/>
   </properties>

  </persistence-unit>

</persistence>
```