FCTUC
UNIVERSIDADE DE COIMBRA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
Departamento de Engenharia Informática

Assignment for Classes
Integração de Sistemas/
Enterprise Application Integration

2019/20 – 1st Semester
MEI, MEIG

Deadline: 2019-11-08

Nota: A fraude denota uma grave falta de ética e constitui um comportamento não admissível num estudante do ensino superior e futuro profissional. Qualquer tentativa de fraude pode levar à reprovação na disciplina tanto do facilitador como do prevaricador.

MUITO IMPORTANTE: o código entregue pelos alunos vai ser submetido a um sistema de deteção de fraudes.

VERY IMPORTANT: the code delivered by students will be submitted to a fraud detection system.

Three-tier Programming with Object-Relational Mapping

Objectives

- Gain familiarity with the development of three-tier enterprise applications using the **Java Enterprise Edition (Java EE)** model.
- This includes the development of applications based on **Enterprise JavaBeans (EJB)**,
- The **Java Persistence API (JPA)** and the **Java Persistence Query Language (JPQL)**.
- Learn to use logging.
- Overall, students should build an Enterprise Archive, which can be deployed on an Application Server.

Final Delivery

- If you are interested in using a technology that is not Java EE you should talk to the Professor to discuss your options. Be prepared to have less support in this case.
- The assignment should be made by groups of two students. Do not forget to associate the colleague during the submission process.
- This assignment contains two parts: one is for training only and does not count for evaluation. Students should only deliver the other part.

- Students must submit their project in a zip file with the **source** of a complete Maven project, using Inforestudante.
- The submission contents are:
    - Source code of the project ready to compile and execute.
    - Project ready to deploy on the application server (an EAR file).
    - A small report in pdf format (6 pages max) about the implementation of the project. See below for details

---

Software

Java EE Platform

To deploy the implementation of this project, students are required to use the **WildFly Application Server**. Students are encouraged to use the latest version of WildFly, although the last successfully used version in this course was 14.

Integrated Development Environment

Eclipse IDE for Java EE Developers and IntelliJ are the recommended IDEs. If students are using Eclipse, they might want to consider the installation of server adapters for WildFly[1], but this step is optional. Students should use **Maven** instead of Eclipse (or IntelliJ) managed projects.

Data Persistence

Object/Relational Mapping (ORM) is a technique to convert data back and forth between object-oriented languages and relational databases. In the case of Java several ORM options exist. In this project, students will have to use Java Persistence API (**JPA**) as it is a Java EE standard. The JPA engine to use should be **Hibernate**. The recommended databases are **MySQL** and **PostgreSQL**, but students are free to choose another one.

References

One of the main references for this assignment is the **course's blog (http://eai-course.blogspot.pt)**. The messages from October 2014, October 2015 and October 2016 let students use JPA in a standalone environment and include notes for using JPA (and EJB) in a container-managed environment. The message from 2018 includes a tutorial to configure a MySQL datasource. The message from 2017 provides a Hello World for Maven. Here are the messages that students should consider:

- https://eai-course.blogspot.com/2018/10/creating-mysql-datasource-in-wildfly-14.html
- https://eai-course.blogspot.com/2017/09/hello-world-maven.html

[1] Check this site, for example: http://www.mastertheboss.com/jboss-server/wildfly-8/configuring-eclipse-to-use-wildfly-8.

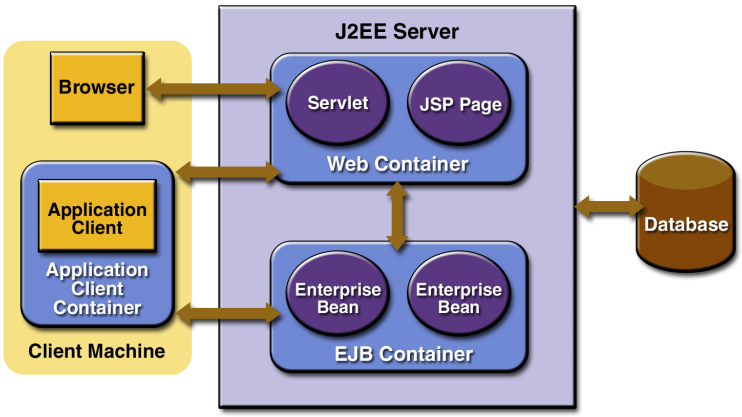
- http://eai-course.blogspot.pt/2014/10/an-enterprise-application-repository-to.html
- http://eai-course.blogspot.pt/2014/10/java-persistence-api-with-eclipselink.html
- http://eai-course.blogspot.pt/2012/10/a-simple-enterprise-javabeans-31.html

At the time of writing of this assignment, there is still no message concerning the creation of a complete Enterprise Application using Maven. To overcome this issue, I published the Maven structure that might serve as the basis for project 2 at Inforestudante. Nonetheless, students might still want to read the available blog messages, which will help them with the exercises.

There are many online resources about Java EE. One of the most important resources is the JavaTM Platform, Enterprise Edition (Java EE) Specification, v8, available at: https://javaee.github.io/javaee-spec/download/JavaEE8_Platform_Spec_FinalRelease.pdf. This document is huge and extremely detailed, but students should skim it, as it covers the whole Java EE.

Introduction to Enterprise Applications

Application Integration can be made across many types of different applications and systems, using a variety of languages and platforms, like Java Enterprise Edition, .NET, Ruby on Rails, (Python) Django, (Python) Flask, and so on. Additionally, traditional three-tier architectures, Service Oriented Architectures and Enterprise Service Buses rely on application servers. Their features, such as support for transactions and security are also very important for standalone microservices. This idea is the basis for WildFly Swarm for example. As a consequence, for the purpose of this course, it is important that students have a deeper contact with the reality of Application Servers.



The idea of Java EE is that enterprise-level applications are structured in three layers: presentation, business logic and data. This is shown in the above image. While the presentation layer is increasingly developed with Javascript frameworks, like VueJS or ReactJS, in this course we are mostly concerned with the other two layers: business and data. These should run in a Java EE application server, which is fully responsible for managing it. The advantage of using an application server, when compared with developing a stand-alone application, is that the programmer does not have to implement many boilerplate enterprise-level functionalities. The container already provides these. In particular, a Java EE server provides for:

- Transactional contexts for guaranteeing data integrity and recovery in case of crashes;
- Connection polling for large number of invocations and optimized access;
- Integrated security management;
- In many cases, distributed computing, load balancing and clustering capabilities.

It should be noted that when one starts using an application server, it gets the feeling that it's slow, cumbersome, and that it has a difficult programming model. It is almost tempting to use a much simpler framework. The advantages only become clear in large-scale contexts, where distributed transactions, transparent load balancing across several servers and millions of invocations have to be made with guaranteed operation. ***In fact, for small-scale applications, Java EE should not be the way to go.*** ☺

Training (not for evaluation)

JPA

1. Create an Entity that stores information about football players. Each player has a name, a date of birth, height and position.

2. Add an option to the program, to list all players of a given position.

3. Add an option to the program, to list all players taller than a given height.

4. Create another entity to store teams. Besides the team name, teams can also have address, president's name, etc. Now, associate players with teams.

5. Finally, students should change the entities and add another option to the program, to allow a connection between players and their former or current team.

6. List all the players that played on a given team.

EJBs

1. Write a stateless EJB that computes the square root of a number. Students should also write a client that uses this bean.

2. Write a stateful EJB that stores and retrieves a list of items given by the client. This EJB could be used to keep a shop basket. Students should also write the client.

3. Write a singleton EJB to raise one alarm. The container should start the EJB automatically. Which solutions could one use to run multiple alarms?

4. Now, write an EJB that allows to use all the functionalities developed in the JPA part.

Project (for evaluation)

Overview

In this project, students will develop a web application to manage an online store of secondhand items, called MyBay. MyBay contains information of users and their items. Next, we go through the requirements of this assignment.

Requirements

1. As an unauthenticated user, I must only have access to the login/register screen.
2. As an unregistered user, I want to create an account, and insert my personal information, including email and country.
3. As a user, I want to authenticate and start a session with my e-mail and password.
4. As a user, I want to logout from any location or screen.
5. As a user, I want to edit my personal information.
6. As a user, I want to delete my account, thus erasing all traces of my existence from the system, including my available items.
7. As a user, I want to put new items for sale. The item should include a name, a category, a country of origin and a photograph.
8. As a user, I want to edit the previous information.
9. As a user, I want to delete an item.
10. As a user, I can list all the items I have for sale sorted by date of insertion.
11. As a user, I want to search for all items.
12. As a user, I want to search for all items inside a single category.
13. As a user, I want to search for all items within a price range.
14. As a user, I want to search for all items in my country.
15. As a user, I want to search for all items published after a given date.
16. The previous searches should allow to include a string to be partially matched against the product name.
17. For all the previous searches, I can sort by name, price or date in ascending or descending order. I can also select each item to see details.
18. The system periodically sends a catalog of items to users by email. This catalog should include the three newest products.

Additional Remarks

This application should have three distinct layers:

- A data layer, working atop a database, to keep all the information (except video files, which might be stored on the server file system). This layer should expose a CRUD (Create, Read, Update, Delete) functionality using EJBs. Internally, it should use Java Persistence API. **In this assignment, the result of the queries must be decided in this layer, i.e., the sorting order, what items to show etc. You should not resort to JavaScript for this purpose.**
- A business layer that interacts with the data layer, built with EJBs. Since the assignment is small, it is acceptable if the business layer uses entities and the

separation of the two layers is not complete.

- A presentation layer developed in some Java-based technology, like JSFs or JSPs. Data between the presentation layer and the business layer **should be transferred using Data Transfer Objects or other techniques** (like sending a simple identifier). Students should not let the entities travel all the way to the presentation layer. For simplicity students might transfer entities between the data and the business layer.

Finally, the following points apply:

- **Passwords should not be stored in clear text**. Students should store hashed passwords in the database.
- **Students must use a logging tool**.
- Students should be very careful about choosing between stateless beans and stateful beans. Also, they should be very careful about not passing huge amounts of information between application layers (e.g, in queries), when they don't need to.
- Students should be careful to avoid violating layers, e.g., they must not perform business logic operations in the presentation layer.
- Students need to be careful about authentication, when they access the EJBs. They must verify each access to protected resources, or otherwise anyone could do a lookup to invoke an EJB.
- Consider the possibility of developing an entire text-based interface, starting only the web tier after the implementation of all the more important functionality. Please note that the weight of the web interface in the final grade is small.

Suggestion of Report

Students may divide the report according to the structure of the application, mentioning in each section the most important aspects, of which I suggest a few of them:

- Introduction
- Presentation Tier
  - Refer to the structure of the webpage, if possible, with a layout (namely templates, if students have used them).
  - Refer to the details that were most complex to implement in the interface.
  - Use a reduced number of images (possibly only one) to show the interface aspect.
  - Focus the aspect of password storage.

- Business Tier
  - Describe and, if necessary, justify the choices they made regarding the type of EJBs (stateless, stateful, interface types, transaction management, etc.).
  - Organize this section by EJBs and their services.
  - Detail the most important aspects. For example, what data is being transferred from the business layer to the presentation (and how).
  - What data do students collect from the data layer to the business layer (pay attention to quantity)?
- Data tier
  - Include ER.
  - Include a (partial) listing of entities.
- Project Management and Packaging
  - Include the description of the package of the program (pom.xml) and structure of the project(s).

Good Work!