CS 548—Fall 2017
Enterprise Software Architecture and Design
Assignment Four—Object Relational Mapping

Provide the data model for a domain-driven design for a clinical information system. You will define a data model using JPA-annotated Java code, and use Eclipse tools to configure database tables for storing the data. In the next assignment, you will add domain-specific logic to the Java classes. **Pay careful attention to the versions of the tools, described below, that you are required to use.**

1. The domain for the clinical information consists of patient entities, for the patients that are treated at that clinic, and provider entities for the health providers that work at that clinic.
2. Each patient entity has a patient identifier, that uniquely identifies them in the health system, a patient name and date of birth, and is related to a collection of treatment entities. A patient should have a database key that is auto-generated by the database.
3. The set of possible treatments, and therefore the possible forms of the treatment entities, is open-ended. Every treatment entity includes a provider administering the treatment, and a diagnosis of the condition for which the treatment is prescribed (e.g. throat cancer, HIV/AIDS, hepatitis, etc). Currently there are three specific forms of treatment entities:
    a. A drug treatment entity includes a drug and a dosage.
    b. A surgery treatment entity includes a date of surgery.
    c. A radiology treatment entity includes a list of dates of radiology treatments. You can define an entity class for each radiology treatment, or you can use the `@ElementCollection` annotation in JPA 2.x to handle a collection of radiology treatment dates.

    Use the join strategy or the one-table-per-class strategy to represent subclass entities in the database.
4. A health provider has a National Provider Identifier (NPI), a name and an indication of their specialization (surgery, radiology, oncology, etc). A provider will be related to patients by the treatments that she performs on them.
5. In both Patient and Provider entities, the list of treatments should be initialized to the empty list.

A provider is not limited to one specific form of treatment. For example, a surgeon may prescribe painkillers or antibiotics as part of post-operative treatment.

We have the following relationship types between entities:
1. There is a one-to-many relationship between patients and treatments.

2. There is a one-to-many relationship between providers and treatments. Each provider is related to each of the treatments for which they are responsible, and therefore they are indirectly related to the patient being treated[1].
3. The deletion of a patient or provider record should cause the deletion of any treatment records related to them.

You should define a collection of Java classes that represent this data model using JPA. Define this collection of classes as an Eclipse Maven project using the Dali plugin. Dali is pre-installed in Eclipse for Java EE. Represent the entity and relationship types using JPA annotations. From this JPA-annotated code, generate the SQL scripts for creating the corresponding tables in a database. Use the Postgresql database that you have installed in the cloud, use the EclipseLink user library, and use the Payara application server as the target runtime. You can find more information about these tools in the Dali tutorial that is provided. You do not need to use the application server for this assignment, you just need these libraries available to Eclipse (on your local machine) while you are editing your JPA classes. You will need to run the Postgresql server to generate the SQL scripts (by connecting to the server from Eclipse), but for this assignment you can connect to a local installation of the Postgresql server to generate the SQL scripts.

Your solutions should be developed for Java 8 and Java EE 7. You should use Payara (Glassfish 4.1) and Eclipselink 2.5.x (Glassfish System Library) . Your project source should be exported from Eclipse Oxygen as a Maven project. In addition, record a short mpeg or Quicktime video of a demonstration of your generation of the database tables from the entity classes (using Eclipse, connected via the JPA perspective to your database server, which can be running locally). Make sure that your name appears at the beginning of the video. For example, display the contents of a file that provides your name. *Do not provide private information such as your email or cwid in the video*. Be careful of any "free" apps that you download to do the recording, there are known cases of such apps containing Trojan horses, including key loggers.

Your solution should be uploaded via the Canvas classroom, as a zip file. This zip file should have the same name as your Canvas userid. It should unzip to a folder with this same name, which should contain the files and subfolders with your submission.

It is important that you provide a document that documents your submission, included as a PDF document in your submission root folder. Name this document README.pdf. As part of your submission, export your Eclipse project to your file system, and then include that folder as part of your archive file. ***Your written report should provide all of the entity classes for your***

[1] You can think of Treatment as a reified relationship type or association class relating Patient and Provider entity types.

submission. The report should also include, in an appendix, the SQL scripts that you generated via Dali for creating the database tables.