**Nov 19(Wednesday)**

1. Discussed Relational Model for the project.
2. Finalized Login Page, Admin Page, Relationship between user –course-role.
3. Finalized the functionality of FOLLOW and LIKE.
4. Finalized page for adding TA and instructor
5. Thought of implementing Email functionality, SP, Indexes and FORM(?)
6. Decided relationship between User, Thread and post.
7. Created TestDAO for User Creation. Validated user creation (able to insert records in the tables)

**Problem:** Unable to decide the relationships between **User-Course-Role** Entities/Classes in JPA. We explored two solutions

Either implement through **SP** (Extract manipulated data directly from DB using SPs) or

**Create a Util** class and then manipulate data in JAVA.

**Solution:**

Discussed with Jose and he proposed using JPAs Entity relationships (we were aware of this solution but we were not sure whether or not to go with this approach http://www.javaworld.com/article/2077819/java-se/understanding-jpa-part-2-relationships-the-jpa-way.html)

**Nov 20 (Thursday)**

1. Configured JPA project in eclipse (i) JPATest , (ii) lms project. And also created tables using JPA and also inserted record into tables

Problem Faced

**Problem:**

**Solution:** mysqldriver(JAR) to Project(lib) was already present , we refreshed the project the project and it worked.

**Problem:** Tables were getting truncated on every execution (Record was getting deleted)

**Solution:**  Commented < property name="eclipselink.ddl-generation" value="drop-and-create-tables" /> from Persistence.xml

**Problem**: Red line below@Entity in All entity classes.

**Solution:** Created corresponding tables

1. Completed set up for Git.
2. Created Entities for User, Course and UserCourseDetail.

**Problem:** Error occurred due to unmapped columns in Entity classes.

**Solution:** Resolved by adding **@Column** annotation above fields in entity.

1. Created UserDAO for User Creation. Validated user creation.

**Problem:**  We were exploring solutions by implementing either SP …………

**Solution: jose proposed solution**

**Problems Faced.**

1. Created DB – lms (Tables -> User, Role, Course, User\_Course\_Detail)

**Nov 21.**

1. Explored different ways of relating Entities in JPA.
2. Could easily implement **one-to-many unidirectional** relationship in Test Project.
3. But we required **one-to-many bidirectional** relationship.

**Problem:** Error was coming due to incorrect syntax for many-to-one and one-to-many relationships.

**Solution:** http://www.javaworld.com/article/2077819/java-se/understanding-jpa-part-2-relationships-the-jpa-way.html

**Problem:** Error occurred while inserting data with one-to-many bi-directional relationship

**Solution:** added (insertable=false and updatable = false) in owing entity (UserCourseDetail)

**Problem:** While debugging we could not view the fetched data due to Lazy loading fetch type.

**Solution:** Update fetch type to EAGER in every Entity, where relationship exists.

**Problem:** StackOverFlow error -> due to infinite mutual recursion between Inverse (Course and User classes) and Owning (UserCourseDetail) entities because of bidirectional one to many relationships. (This error occurred while implementing .toString())

**Solution:** In .toString we added only the required fields (instead of User , we put user.getFirstName and user.getLastName).

**Problem:** JPA OneToMany - List doesn't update after add/delete Object, but persist in DB. Due to which we were getting NULL values for recently inserted records.

**Solution:** added  **em.refresh()** after **em.getTransaction().commit(); in** UserCourseDetailDAO.createUserCourseDetail

**Nov 22**

**Traget : Complete CRUD operation for USER, COURSE AND USERCOURSEDETAIL.**