**Hands on 2**

**Hibernate XML Config implementation walk through**

* **object to relational database mapping done in hibernate xml configuration file:**

In Hibernate XML configuration, Java objects are mapped to database tables using a mapping file (usually named as \*.hbm.xml). This is a key part of Hibernate's Object-Relational Mapping (ORM) approach.

Ex: (Employee.hbm.xml)

<class name="com.example.Employee" table="employee">

<id name="id" column="id">

<generator class="increment"/>

</id>

<property name="name" column="name"/>

</class>

<class> - maps a Java class to a DB table

<id> - maps the primary key

<property> - maps the Java fields to DB columns

generator - defines how the primary key is auto-generated

* **SessionFactory:**

A thread-safe, heavyweight object created once during application startup. It is configured using hibernate.cfg.xml file, used to create Session objects.

Ex:

SessionFactory factory = new Configuration()

.configure("hibernate.cfg.xml")

.addResource("Employee.hbm.xml")

.buildSessionFactory();

* **Session:**

Represents a single-threaded unit of work with the database. It manages CRUD operations for persistent objects.

Ex:

Session session = factory.openSession();

* **Transaction:**

Used to group multiple operations into a single atomic unit. Ensures data integrity (ACID properties).

Ex:

Transaction tx = session.beginTransaction();

* **beginTransaction():**

Begins a new database transaction. Required before performing save/update/delete operations.

Ex:

tx = session.beginTransaction();

* **commit():**

Commits the transaction and persists changes to the DB.

Ex:

tx.commit();

* **rollback():**

Rolls back the transaction if an error occurs, undoing all changes.

Ex:

tx.rollback();

* **session.save():**

Saves a new object into the database and returns the generated ID.

Ex:

Employee emp = new Employee("John", "Doe", 50000);

session.save(emp);

* **session.createquery().list():**

Used to fetch a list of objects using HQL (Hibernate Query Language).

Ex:

List<Employee> employees = session.createQuery("FROM Employee").list();

* **session.get():**

Fetches an object by its primary key. Returns null if not found.

Ex:

Employee emp = session.get(Employee.class, 1);

* **session.delete():**

Deletes the given object from the database.

Ex:

session.delete(emp);