Lab03 - Accessing databases through JDBC

[IT618, Enterprise Computing, Autumn'23]

Instructor: PM Jat (pm_jat@daiict.ac.in)

In this lab, let us extend our inventory application of Lab01 and make it database persistent.

Let us use relational databases, and have the following two tables for this application.

1. Table `Category` with attributes: `cat_id int, cat_name varchar(50)` for storing item categories.
2. Table `Item` with attributes: `code int, description varchar(50), stock int, min_stock int, cost decimal(8,2), cat_id int)` Here `cat_id` is foreign key referring to `Category`

Create these tables in DBMS that you use and put some sample rows in each tables. Put around 10 item of 3 categories.

For implementation, suppose, we have identified the following interface for Data Access Object for Item `ItemDAO` as discussed in lectures.

```
public interface ItemDAO {

   public void add(ItemTO item) throws DAOException;
   public void update(ItemTO item)
           throws ItemNotFound, DAOException;
   public ItemTO findItem(int item_code)
           throws ItemNotFound, DAOException;
   public void delete(int item_code)
           throws ItemNotFound, DAOException;
   public ArrayList<ItemTO> getAll()
           throws DAOException;
   public ArrayList<ItemTO> getAllPaginated(int page_no)
           throws DAOException;
}
```

Let you do the following in this Lab

1. Create a concrete class `ItemDAOImplSQL` that implements the above methods.

   For your reference here is a `BookDAOImplSQL` https://github.com/pmjat/j2ee/blob/master/j2ee/book/dao/sql/BookDAOImplSQL.java
2. Create a console-based Tester class that performs some of CRUD operations on the `Item` table through said DAO Implementation.
3. Let the `Inventory` class be named as `InventoryService` class. Say following is interface for `InventoryService`. Implement the service class using `ItemDAO` objects.

```
    ItemTO getItem(int itemno) throws ItemNotFound
    //returns new object of inventory item after reading data
    //     for given item number from database
    //     Throws exception if item not found.
```

```
    void addItem(ItemTO item) throws ItemAlreadyExists
    void updateItem(ItemTO item) throws ItemNotFound
    void addStock(int item_code, int qty) throws ItemNotFound
    void withdrawStock(int item_code, int qty)
     throws ItemNotFound, InSufficientStock
    void deleteItem(int item_code) throws ItemNotFound
    ArrayList<ItemTO> getAllItems() //returns all items in inventory.
    ArrayList<ItemTO> getAllCatItems(int cat_id)
            //returns all items in inventory.
    ArrayList<ItemTO> getItemsUnderStock()
    //returns all items that are under stock,
    //      i.e. below required minimum stock.
    double totalInventoryCost()
    //returns total cost of inventory,
    //      i.e. summation of cost of all items in the inventory
```

4. Also create a Inventory Service tester class.

PUT everything in package `jee.lab03`

Deliverables of this exercise: Source Code of `ItemDAOSQLImp`, `ItemDAOTester` `InventoryService`, and `InventoryTester`