


ENSF 607 – Software Design and Architecture I
ENSF 608 – Software Design and Architecture II
Fall 2020

Cross-Course Project

ENSF 607 Due Dates	
Pre-Project Exercise and UML Diagram	Submit electronically on D2L before 11:59 PM on **Wednesday November 18th.** **Only one** group member should submit the project on D2L but **put both names on the submission.**
Final Project	Submit electronically on D2L before 11:59 PM on **Thursday November 26th.** **Only one** group member should submit the project on D2L but **put both names on the submission.** **Demo: You must demo your project to your TA during the lab on Friday November 27th.**

ENSF 608 Due Dates	
Conceptual & Logical Designs	Submit electronically on D2L before 11:59 PM on **Wednesday November 18th.** **Only one** group member should submit the project on D2L but **put both names on the submission.**
Database Video Demo	Submit electronically on D2L before 11:59 PM on **Friday, November 27th.** **Only one** group member should submit the project on D2L but **put both names on the submission.** **You will not need to demo during a lab session.**


This is a **group** assignment. Students may work in groups of two. You can pair up regardless of your TA groups. Please note, students have the option of working on their own.

Important Note: Even though this is a group assignment, it is **very important** that both people in the group work on all exercises and collaborate closely together. This is a great opportunity to gain valuable pair programming (https://www.agilealliance.org/glossary/pairing/) experience and engage in collaborative design. Taking the approach of "I do this part, and you do the next" is not a good idea and simply results in a missed opportunity!

The objectives of this project are:

1. Conceptual database design based on a narrative
2. Logical database design through relational model mapping
3. Implementation and querying of a database
4. Java Graphical User Interface Event-Handling
5. Client-Server Architecture
6. Working with database through a Java program
7. MVC

The following rules apply to this project and all other assignments in future:

1. Before submitting your project reports, take a moment to make sure that you are handing in all the material that is required. If you forget to hand something in, that is your fault; you can't use `I forgot' as an excuse to hand in parts of the assignment late.
2. **20% marks** will be deducted from the assignments handed in up to **24 hours** after each due date. It means if your mark is X out of Y, you will only gain 0.8 times X. There will be no credit for assignments turned in later than 24 hours after the due dates; they will be returned unmarked.

Pre-Project Exercise: Accessing a MySQL Database with Java (10 Marks)

For this exercise, you need to download `InventoryManager.java`, `Tool.java`, and `ItemsNew.txt` from D2L, then install MySQL and JDBC on your computer. Detailed instructions for MySQL are posted on D2L under "Some Help with MySQL".



What to do:

Task 1 – Create an empty database on your computer with MySQL. Then, make any necessary changes to the `InventoryManager` class to make it connect to that database and run without error. You will need to provide your username, password, and a valid path to the database to connect. If you are successful, your program output should look like this:

```
Connected to: jdbc:mysql://localhost:3306/InventoryDB

Created Table ToolTable

Filling the table with tools
Reading all tools from the table:
Tools:
1000 Knock Bits 88 12.67 8015
1001 Widgets 10 35.5 8004
1002 Grommets 20 23.45 8001
       .
       .
       .
1038 Googly Eyes 756 6.99 8001
1039 Handy Pandies 321 4.35 8017
1040 Inny Outies 219 3.45 8010

Searching table for tool 1002: should return 'Grommets'
Search Result: 1002 Grommets 20 23.45 8001

Searching table for tool 9000: should fail to find a tool
Search Failed to find a tool matching ID: 9000

Trying to remove the table
Removed Table ToolTable

The program is finished running
```

Hint: There are many ways to create a database using command line, MySQL Workbench or even a Java program, but the database only needs to be created once. Once created, you should be able to connect to it with `InventoryManager` by providing the correct path.

Task 2 - Next, your job is to replace the `Statement` object in `InventoryManager` with a `PreparedStatement`. You will need to modify each method accessing the database to make the program function as it did before.

Note: For full marks in this exercise, wildcard characters (?) should be used in the `addItem` and `searchTool` methods.


What to Submit: Please submit all the java files including the files you have created/modified in a zip folder. You need to provide the Javadoc comments in your code, but you don't need to submit the HTML files.


Project – Design and Development of a Distributed Inventory Management System Using MVC (90 Marks + 10 Possible bonus Marks)

In this project, you will redesign the inventory management system into a distributed system using the client-server architectural pattern. Your design must follow the MVC design pattern.

Recall the original software requirement:

Requirement description - A small retail shop that sells tools requires an application to manage inventory of different types of tools it sells. The store owner wants to be able to modify the store's inventory by adding new tools, and deleting tools. The owner also wants to be able to search the inventory for tools by tool name, and by tool id. Currently, the information about tools available in the shop and suppliers is stored in two text files which are given on D2L: `items.txt`, and `suppliers.txt`.

The order and type of data given in these files are:

`items.txt`:

`(id; tool type; description or name of tool; quantity in stock; price; supplier id number)`

`Suppliers.txt`:

`(id; supplier type; company name; address; sales contact)`

The owner would also like to check the quantity of each item in stock. If the quantity of each item in stock goes below 40 items, then the program should automatically generate an order line for that item. The order line will have the supplier information and the required quantity for that item (The default quantity ordered by each item = 50 – number of existing items). All items ordered each day should be included in an order which has a randomly generated 5-digit id, and the date that was ordered. The order should be written to a text file called `orders.txt`.



You will add the following functionalities for the system:

- There are 2 types of suppliers:
  - International suppliers
    - Have an import tax field and a method to calculate it (you don't have to implement that method)
  - Local suppliers
- There are 2 types of items:
  - Electrical
    - Have information about power type.
  - Non-Electrical

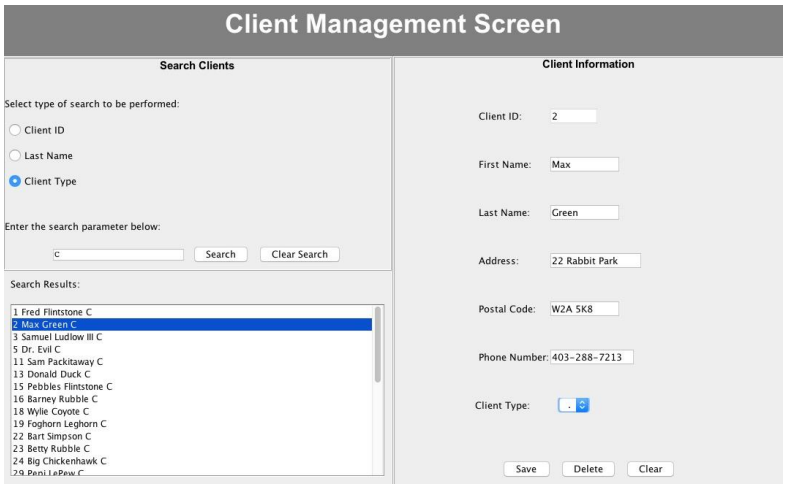
You will also add a customer-management (also known as client management) component to this system.

A client has an ID which is a unique number assigned to the client when defining a new customer. Firstname, Lastname, and the address of the client can have a maximum length of 20, 20, and 50 characters respectively. The maximum length of the postal code and phone number fields are 7, and 12 characters respectively. A client can be of two types: Residential (R), or Commercial (C).

The customer management system will support the following functionalities:

- Add a new customer (i.e. client)
- Modify existing information for a customer
- Remove a customer from the database
- Search for a customer based on id number, name, or type.

A sample GUI is shown in the following image (this is just an example; you are free to design your GUI differently). In the left panel, the user can search for a client based on three criteria. The results of the search are shown on the bottom left side of the window. This list is created using a `JScrollPane`. The items of the list have a listener which enables us to click on them to see the detailed information of the customer in the right window. On the right side of the window, we can modify an existing client, or delete it. We can also add new clients.



The system will have at least 2 GUIs:

1- Inventory management system. The functionality of the inventory management system is exactly the same as the console-based menu you implemented in an earlier lab:
   a. List all tools (this must be handled by the proper toString methods in the backend).
   b. Search for tool by toolName
   c. Search for tool by toolID
   d. Check item quantity
   e. Decrease item quantity (This is to simulate a sale of the item. Once the item count goes under 40, this function should trigger the creation of an order).

2- Customer management system (i.e. client management system) as shown above.

ENSF 607 Deliverables:

- UML diagram of the entire system
  - You may want to have a one high-level and a few lower level diagrams
- Completed development of the system including an MVC client-serve application. Your application must be able to handle multiple clients.

Bonus Marks:

- Implement a simple user-management/login system
- Deploy your client-server application on multiple machines (you can either deploy on multiple computers, or you can deploy your server a cloud server such as Amazon AWS).

How to submit: Include all your files for this lab in one folder, zip your folder and upload it in D2L before the deadline.


ENSF 608 Deliverables (25% of 608 course grade)

Conceptual Database Design (20 marks, 40% of 608 project grade, 10% of course):
Based on the provided requirements narrative, design and draw an EER diagram for the described database application. Your solution should follow the model notation presented in class and should include cardinality ratios and participation constraints. Your diagram should also be accompanied by a half-page description explaining your design decisions and any assumptions that were made.

Your solution may be handwritten or typed, and you may draw your diagram by hand or by using software tools. Handwritten work may be scanned or photographed (tip: try using an app such as Microsoft Office Lens). Marks will be deducted for incorrect or missing information based on the provided narrative. Solutions must be neat and organized.

Logical Database Design (10 marks, 20% of 608 project grade, 5% of course):
Map your conceptual schema into a relational data model, including all primary keys and referential integrity constraints (foreign keys).

Your solution may be handwritten or typed, and you may draw your diagram by hand or by using software tools. Handwritten work may be scanned or photographed (tip: try using an app such as Microsoft Office Lens). Marks will be deducted for incorrect or missing information based on your EER diagram design. Solutions must be neat and organized.

MySQL Video Demonstration (20 marks, 40% of 608 project grade, 10% of course):
Implement your database in MySQL Workbench. Use Zoom or another tool of your choosing to record a **short** demonstration of your database (5 minutes or less!). If you are working with a partner, both students should be part of the demonstration.

In the demonstration, show the following elements using your inventory database:
1) Show all tables and explain how they are related to one another (keys, triggers, etc.)
2) A basic retrieval query
3) A retrieval query with ordered results
4) A nested retrieval query
5) A retrieval query using joined tables
6) An update operation with any necessary triggers
7) A deletion operation with any necessary triggers

Marks will be deducted for incorrect or missing information and may also be deducted for videos of excessive length. Videos should have clear audio and cameras should be on. This is your opportunity to demonstrate how your database works and why it is a correct solution.