



Southern Luzon Technological College Foundation, Inc.



NC III – Java Programming

Project Documentation

GAME INVENTORY IN MYSQL DATABASE AND JSON FILE

Chris Lemuel D.G. Dela Cruz

Student

Benz Vincent A. Geraldizo

Instructor

SUMMARY

This project is a Command-Line Interfaces Game Inventory written in Java programming with basic CRUD functionality in MYSQL database as the game storage box and a Json file acting as the (adventurers) bag. The reason for doing this project is to show what the student learned in the past months of online training as well as incorporating some of what the student learned outside of the training.

As mentioned earlier, this project is written in Java programming language with the help of IntelliJ Integrated Development Environment, Java Development Kit version 24, and with the external Java Archive (JAR) files namely Java Database Connector (JDBC) for connecting in MySQL Database and Jackson (core, annotation, data-bind) for reading and writing Json file. And lastly a MySQL server as for the project database.

PROJECT STRUCTURE

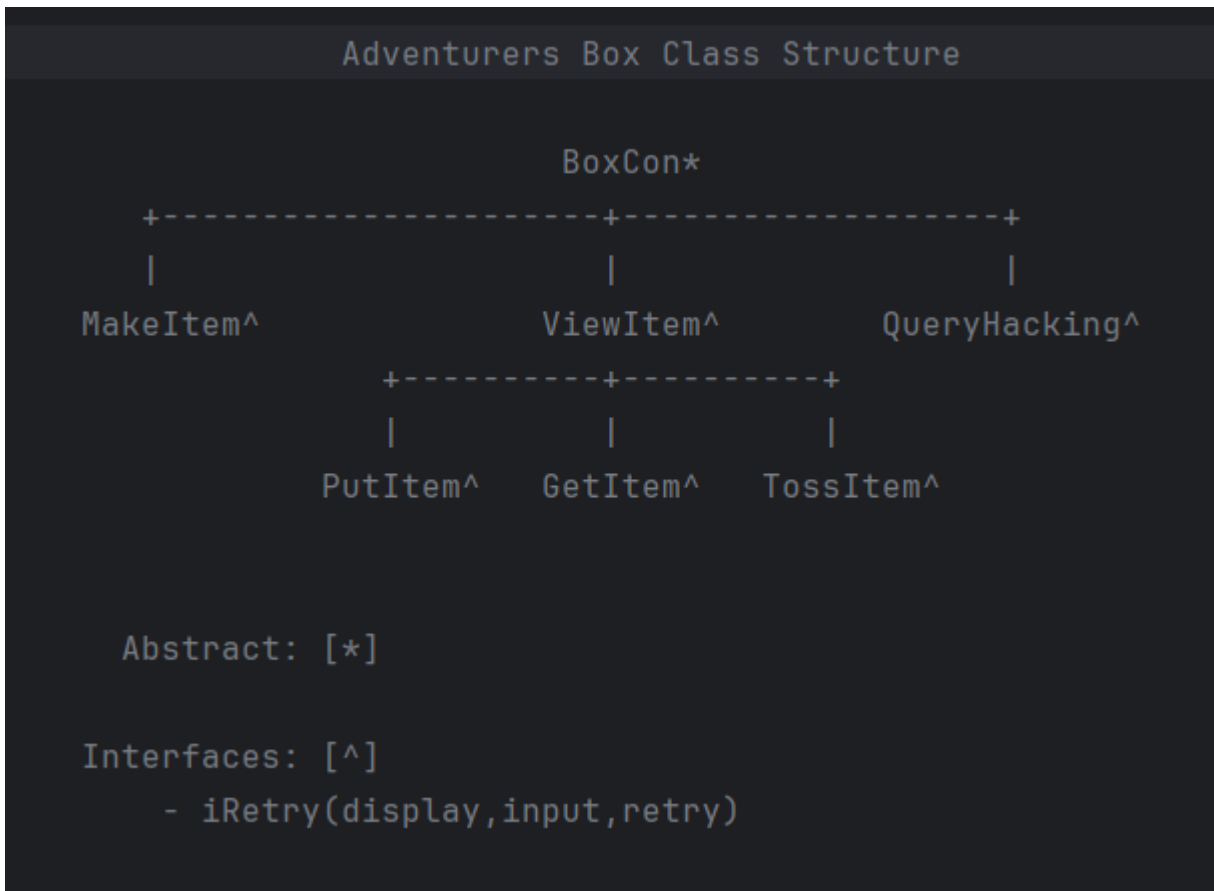


Image 1: Image representation of project structure

The following will be a walkthrough in all the classes and alike that is used for the project named Adventurers Box as a representation for a game inventory. This also includes some classes that are not visible in the image above (Image 1) and will be discussed next page.

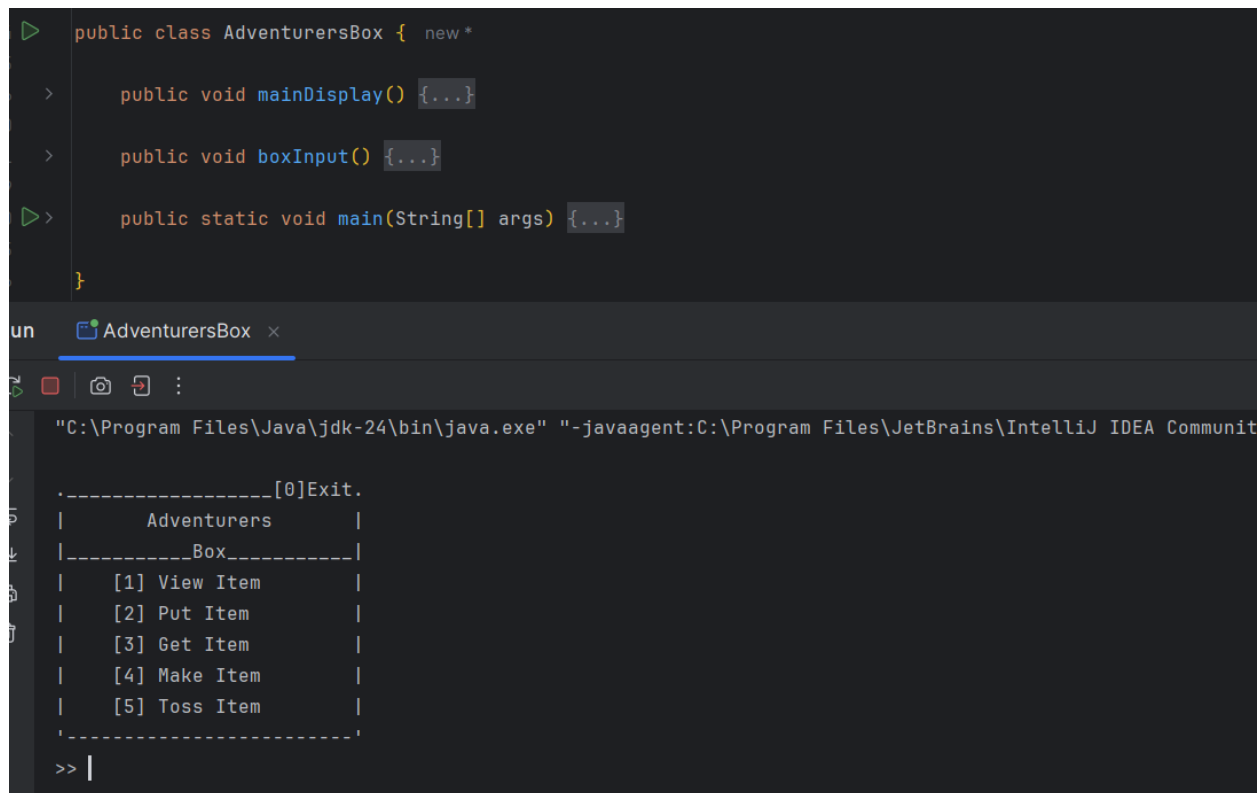
```

3      public class Item { 30 usages  new *
4
5          private int id, amount; 3 usages
6          private String name; 4 usages
7
8      >      public Item(int id, String name, int amount) {...}
13
14      >      public Item(String name, int amount) {...}
18
19      >      public Item() {...}
24
25      >      public void setAmount(int amount) { this.amount = amount; }
28
29      >      public int getId() { return this.id; }
32
33      >      public String getName() { return this.name; }
36
37      >      public int getAmount() { return this.amount; }
40
41      }

```

Item Class

This class contains a representation of an item consisting of three fields for id, name and amount it holds. The class has various constructors for creating items and functions for retrieving or updating the fields. The student purposely put this class in private to demonstrate encapsulation and polymorphism through constructors.



```
public class AdventurersBox { new *
    >     public void mainDisplay() {...}
    >     public void boxInput() {...}
    >     public static void main(String[] args) {...}
}
```

run **AdventurersBox** x

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Communit
.....[0]Exit.
|      Adventurers      |
|-----Box-----|
|  [1] View Item  |
|  [2] Put Item   |
|  [3] Get Item   |
|  [4] Make Item  |
|  [5] Toss Item  |
|-----|
>> |
```

AdventurersBox Class

This class is mainly for executing the program, the main display and input of it. A switch case is made for accessing the other classes and entering zero will exit the program.

BoxCon Class

This class is created for the sole purpose of connecting the program into database and being the base class of the program, as such the student decided that making it an abstract class since the other classes would be only inheriting its functions and fields.

iRetry Interface

This interface holds three methods namely display, input and retry; it is implement for the remaining classes in this project, it's pretty self-explanatory for what it does.

```
5 public class MakeItem extends BoxCon implements iRetry{ 2 usages Remyueru-AI
6
7 @Override Remyueru-AI
8 public void display() {...}
12
13 @Override 9 usages Remyueru-AI
14 public void input() {...}
33
34 public void confirmMake(String name,int amount) {...}
48
49 public void insert2Box(String name,int amount) {...}
79
80 @Override 22 usages Remyueru-AI
81 public void retry() {...}
```

Run AdventurersBox x

```
↑ [2] Put Item |
↓ [3] Get Item |
⇅ [4] Make Item |
⇅ [5] Toss Item |
-----
>> 4

Adventurers Box > Make Item
Enter Name: Test
Enter Amount: 2

Adventurers Box > Make Item > Confirm
Make 2x of Test? [Y][N]
>> |
```

MakeItem Class

This class is used for two things, adding a new item to the database or adding an existing item amount if item name matched. It is done in the [insert2Box](#) function that will search and alter the amount for the item if it has already existed, otherwise it will become a new item in the database, with that it solves the problem of duplicating an existing item.

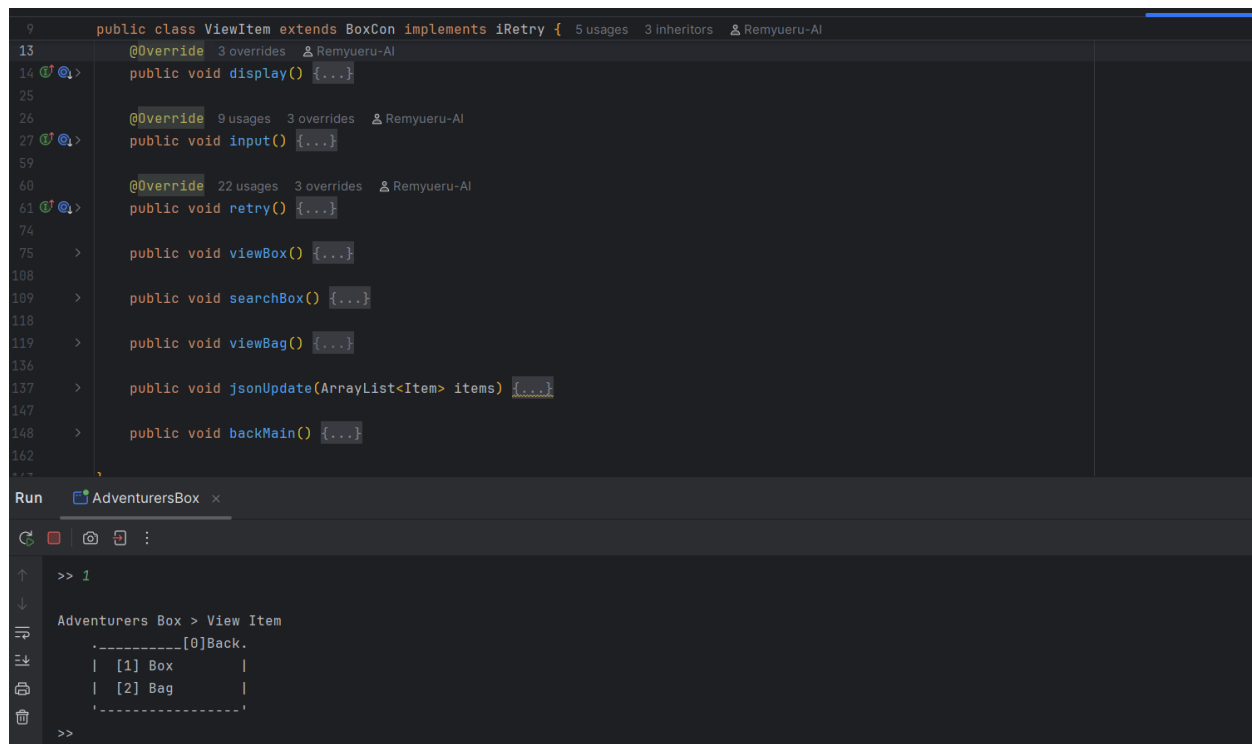
```

        public void insert2Box(String name,int amount) {
System.out.println("Processing...");
qry = "select * from boxTb where itemName like '" + name + "'";

try {
    connectBox();
    statement = con.createStatement();
    resultSet = statement.executeQuery(qry);

    if (!resultSet.isBeforeFirst()) {
        qry = "insert into boxTb values (?,?,?)";
        connectBox();
        preStatement = con.prepareStatement(qry);
        preStatement.setInt(1, 0);//id temp
        preStatement.setString(2, name);
        preStatement.setInt(3, amount);
    }
    else {
        qry = "UPDATE boxTb SET amount = amount + " + amount + " WHERE
itemName = '" + name + "'";
        connectBox();
        preStatement = con.prepareStatement(qry);
    }
    preStatement.executeUpdate();
    System.out.println("The item is in the box\n");
    retry();
}
catch (Exception e) {
    System.out.println("Something went wrong" + e);
}
}

```



```
9 public class ViewItem extends BoxCon implements iRetry { 5 usages 3 inheritors Remyueru-AI
13 @Override 3 overrides Remyueru-AI
14 public void display() {...}
25
26 @Override 9 usages 3 overrides Remyueru-AI
27 public void input() {...}
59
60 @Override 22 usages 3 overrides Remyueru-AI
61 public void retry() {...}
74
75 public void viewBox() {...}
108
109 public void searchBox() {...}
118
119 public void viewBag() {...}
136
137 public void jsonUpdate(ArrayList<Item> items) {...}
147
148 public void backMain() {...}
162
Run AdventurersBox x
>> 1
Adventurers Box > View Item
-----[0]Back.
| [1] Box |
| [2] Bag |
'-----'
>>
```

ViewItem Class

This class has a lot going on inside it, first as the named suggest, it will view items in box OR in bag however the search function is box exclusive for the reason being that the Json bag that the student made is limited to 5 slots. This class is vital because it is useful to know what inventory have OR what the user got in it, furthermore the viewBox function constructs an item based on the first item in the row using Item class constructor, it is made this way for GetItem and PutItem class to have a ready to use item object if **viewBox** is called and acts as an item search for the said classes if SQL query only specified one item. As for the bag it has jsonUpdate for removing Json attributes if the item amount is zero


```

        public void viewBox() {
try {
    connectBox();

    statement = con.createStatement();
    resultSet = statement.executeQuery(qry);

    if (!resultSet.isBeforeFirst()) {
        System.out.println("""
            +-----+
            | NO ITEM FOUND |
            +-----+""");
        myItem = new Item();
    }
    else {
        while (resultSet.next()) {
            System.out.printf("+-----+\\n|\\t%-10s\\t| %-2d \\n",
                resultSet.getString("itemName"), resultSet.getInt("amount"));
            if (resultSet.isFirst()) {
                myItem = new Item(
                    resultSet.getInt("itemId"),
                    resultSet.getString("itemName"),
                    resultSet.getInt("amount"));
            }
        }
        System.out.println("+-----+");
    }

} catch (Exception e) {
    System.out.println("Error: Box not found" + e);
    display();
}
}

```

```
AdventurersBox x
+-----+
| Potion | 14 |
+-----+
| Soda Pop | 10 |
+-----+
| Macca | 3 |
+-----+
| Paint | 9 |
+-----+
| Aeos | 40 |
+-----+
| Photon | 2 |
+-----+
| Herb | 5 |
+-----+
| TM Dive | 1 |
+-----+
| Enigma | 1 |
+-----+
| KEA | 1 |
+-----+

Adventurers Box > View
Search item? [Y][N]
>>
Process finished with e
```

viewBox()

```
Adventurers Box > View Item > Box > Search
+-----+
| KEA | 1 |
+-----+

Adventurers Box > View Item > Box
Search item? [Y][N]
>> y

Adventurers Box > View Item > Box > Search
Item name: 0

Adventurers Box > View Item > Box > Search
+-----+
| Potion | 14 |
+-----+
| Soda Pop | 10 |
+-----+
| Aeos | 40 |
+-----+
| Photon | 2 |
+-----+

viewBag()
```

```
Adventurers Box > View Item
.-----[0]Back.
| [1] Box |
| [2] Bag |
'-----'

>> 2

Adventurers Box > View Item > Bag [5/5]
/-----+-----\
| Paint | 6 |
| Herb | 2 |
| Soda Pop | 2 |
| Potion | 11 |
| Aeos | 15 |
\-----+-----/
Back to main? [Y][N]
>> |
```

```
Adventurers Box > View Item > Box > Search
+-----+
| KEA | 1 |
+-----+

Adventurers Box > View Item > Box
Search item? [Y][N]
>> y

Adventurers Box > View Item > Box > Search
Item name: 0

Adventurers Box > View Item > Box > Search
+-----+
| Potion | 14 |
+-----+
| Soda Pop | 10 |
+-----+
| Aeos | 40 |
+-----+
| Photon | 2 |
+-----+

Adventurers Box > View Item > Box
Search item? [Y][N]
>>
```

searchBox()

GetItem and PutItem Class

This class is almost the same as the other with a little difference that GetItem checks the bag slots and verify if it's at max capacity, it can only add an existing item in the bag via **putItemToBag** function, other than that both are the identical but reversed on what it does in the database and in the Json file.

```
public void putItemToBag(int put, int id, String name) {  
  
    try {  
        ObjectMapper om = new ObjectMapper();  
        items = om.readValue(bagJson, new TypeReference<>() {});  
        Item pcs = new Item(id, name, put);  
        boolean isNewItem = true;  
  
        for (Item i : items) {  
            if (i.getId() == pcs.getId()) {  
                put = put + i.getAmount();  
                pcs.setAmount(put);  
                items.remove(i);  
                items.add(pcs);  
                jsonUpdate(items);  
                isNewItem = false;  
                break;  
            }  
        }  
  
        if (isNewItem) {  
            if (items.size() < 5) {  
                items.add(pcs);  
                jsonUpdate(items);  
            }  
            else {  
                System.out.println("The bag is already full..");  
                retry();  
            }  
        }  
  
    } catch (Exception e) {  
        System.out.println("Error: " + e);  
    }  
}
```

```
Adventurers Box > Get Item      [0] Back  
Enter item name: aeos  
+-----+  
|  Aeos      | 40 |  
+-----+  
  
Adventurers Box > Get Item > Aeos  
Enter amount, you have 40 in storage  
>> 2  
  
Adventurers Box > Get Item > Confirm  
Get 2 Aeos? [Y][N]  
>> y  
  
Adventurers Box > Get Item > Bag [5/5]  
/-----+-----\  
|  Paint     |  6  |  
|  Herb      |  2  |  
|  Soda Pop  |  2  |  
|  Potion    | 11  |  
|  Aeos      | 17  |  
\-----+-----/  
You got x2 Aeos from the box..  
  
Adventurers Box > Get Item  
Back to main? [Y][N]  
>>
```

TossItem Class

While creating or adding an item can only be done in the database, removing items can only be done in the Json file which is acting as an adventurer bag. It isn't difficult to make because it is an ArrayList of Item objects and it pales in comparison to how challenging ViewItem is.

```
|  Aeos      |  15  |  
|-----+-----|  
  
Adventurers Box > Toss Item      [0] Back  
Enter item name: aeos  
  
Adventurers Box > Toss Item > Aeos  
Enter amount, you have 15 in bag  
>> 2  
  
Adventurers Box > Toss Item > Confirm  
Throw 2 Aeos? [Y][N]  
>> y  
  
Adventurers Box > Toss Item > Bag [5/5]  
/-----+-----\  
|  Paint      |   6  |  
|  Herb       |   2  |  
|  Soda Pop   |   2  |  
|  Potion     |  11  |  
|  Aeos       |  13  |  
|-----+-----|  
You threw x2 Aeos away..  
  
Adventurers Box > Toss Item  
Back to main? [Y][N]  
>>
```

QueryHacking Class

This class is hidden because as the name suggested, this will alter anything, even the database itself assuming the query written is correct; it is only included for a little humor making this project. It can be access by typing 143 on the main display. The number choice 143 is references on the ILOVEYOU virus.

```

.-----[0]Exit.
|      Adventurers      |
|-----Box-----|
|  [1] View Item      |
|  [2] Put Item       |
|  [3] Get Item       |
|  [4] Make Item      |
|  [5] Toss Item      |
|-----|
>> 143

Adventurers Box > 143H@k3d
Use Query to hack the box
>>
```

“For anyone want to check the project, it is available in GitHub, I liked to thank my parents, SL pasig staff and sir benz for this to be possible”

- Chris Lemuel D.G. Dela Cruz

GitHub: <https://github.com/Remyueru-AI/Learning-Java/tree/master/src/ProjectJava>