

# APPs Project

EECS 3421 – Introduction to Database Management Sysytems.

Rajkumar Balakrishnan Lakshmi - 213141197

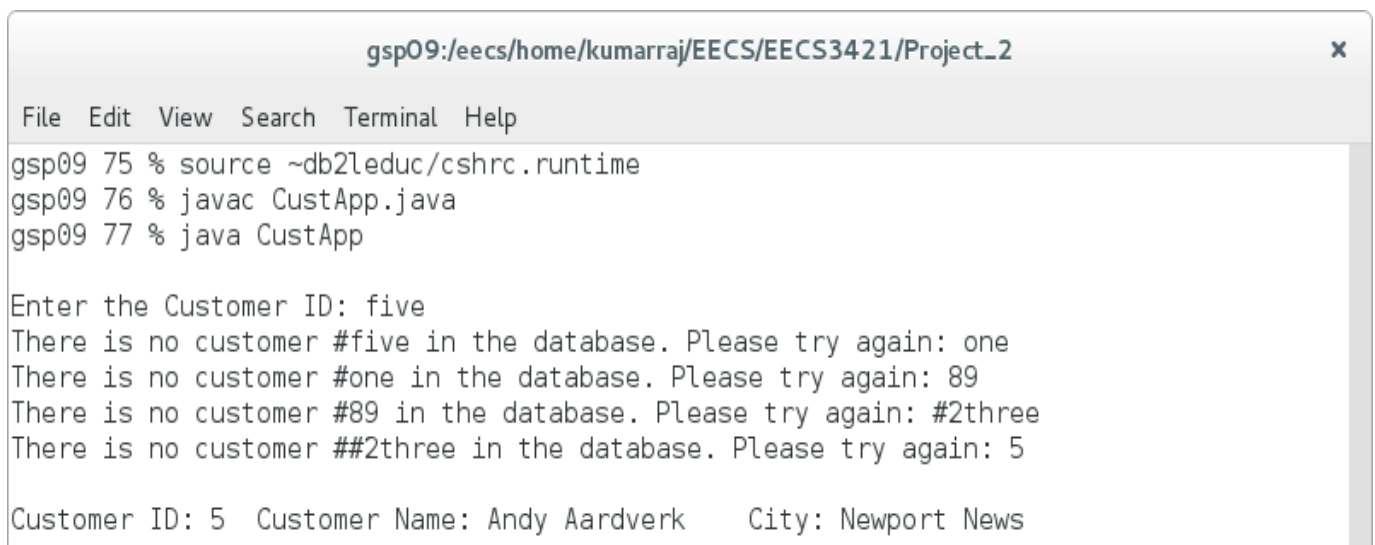
5/12/17

## Report

The aim of this project is to create an application program for the YRB database and the main purpose of this application is to search through the database and try to purchase books that are available on the database.

The program begins by asking the user to input the Customer id and if the entered id exists in the database it displays the details of that customer. Whereas if the entered id is not available on the database or if the user inputs some invalid type values or characters, then it prints an error message and prompts the user to input the customer id again. This process keeps looping through until a valid customer id is entered by the user. A glimpse of this process is shown below:

Input example for – Enter the customer ID: Integer value (1 to n) n- number of customers in the database.



```
gsp09:/eecs/home/kumarraj/EECS/EECS3421/Project_2
File Edit View Search Terminal Help
gsp09 75 % source ~db2leduc/cshrc.runtime
gsp09 76 % javac CustApp.java
gsp09 77 % java CustApp

Enter the Customer ID: five
There is no customer #five in the database. Please try again: one
There is no customer #one in the database. Please try again: 89
There is no customer #89 in the database. Please try again: #2three
There is no customer ##2three in the database. Please try again: 5

Customer ID: 5 Customer Name: Andy Aardverk City: Newport News
```

Now it asks if the user wants to update the information of the displayed customer and if the customer agrees, all possible ways of updating the information is displayed and asks for the user to choose from one of the options. Here again if the user inputs an invalid type of characters or value. It prints an error message and prompts the user to input the option again. This process keeps looping through until a valid option is chosen and based on the option chosen the update is performed. A glimpse of the above process is shown below:

## Input example for – Updating Customer Information: String (Name | City | Both)



```

gsp09:/eecs/home/kumarraj/EECS/EECS3421/Project_2
File Edit View Search Terminal Help
gsp09 79 % java CustApp
Enter the Customer ID: 5
Customer ID: 5   Customer Name: Andy Aardverk   City: Newport News
Would you like to update the customer information? Enter Yes || No : s
The entered option is invalid. Please try again: yup
The entered option is invalid. Please try again: #01
The entered option is invalid. Please try again: yes
Possible ways of updating customer information are:
1.Name
2.City
3.Both
Choose from one of the above options:nmaecity
The entered option is invalid. Please try again: name&city
The entered option is invalid. Please try again: 3
The entered option is invalid. Please try again: both
Enter the new name: Jarek Gryz
Enter the new city: Toronto
Name has been successfully updated to Jarek Gryz !!!
City has been successfully updated to Toronto !!!

```

After updating the customer information, the program displays a list of all categories of books available on the database and asks the user to input or to choose a category from the list to see all the titles available under that category. If a right category is chosen from one of the categories in the list, then the book titles associated with that title is displayed, otherwise if an invalid type or any other invalid option is entered by the user, the program prints an error message and asks the user to input the category again. This process keeps looping through until a valid option is entered.

Now if the user chooses the right category and all the book titles associated with it is displayed, then the program asks for the user to enter a book title from the list of books displayed to see the details of that book. Here if the user enters a book that is not in the list or if the user enters any invalid characters or values, then the program displays an error message and takes back the user to the previous step of choosing the categories again. We assume that the user is looking for a book in a different category, so we go back to the previous step. This routine continues until the user chooses a right book title associated to the right category. A glimpse of the above two processes is shown below:

Input example for – Choosing Category: String (children | cooking | .... | travel)

Input example for – Choosing Book title: String (Database aren't | ... | Plato Rocks)

Note: For the above don't enter a integer value, the input should be the category name or title name only.

```

gsp09:/eecs/home/kumarraj/EECS/EECS3421/Project_2
File Edit View Search Terminal Help
The categories of books in our database are:
1.children
2.cooking
3.drama
4.guide
5.history
6.horror
7.humor
8.mystery
9.phil
10.romance
11.science
12.travel
Choose from one of the above categories to see the book titles: mystery

The book titles under the category "mystery" are:
1.Rigor Mortis
2.The Wiley Snark
3.Under Your Bed
4.Where are my Socks?
5.Will Snoopy find Lucy?
Choose from one of the above book titles to view its details: Plato Rocks

The book "Plato Rocks" under the category "mystery" doesn't exist. Please try again!

-----
The categories of books in our database are:
1.children
2.cooking
3.drama
4.guide
5.history
6.horror
7.humor
8.mystery
9.phil
10.romance
11.science
12.travel
Choose from one of the above categories to see the book titles: phil

The book titles under the category "phil" are:
1.Databases aren't
2.Is Math is fun?
3.Plato Rocks
Choose from one of the above book titles to view its details: Plato Rocks

-----
-----

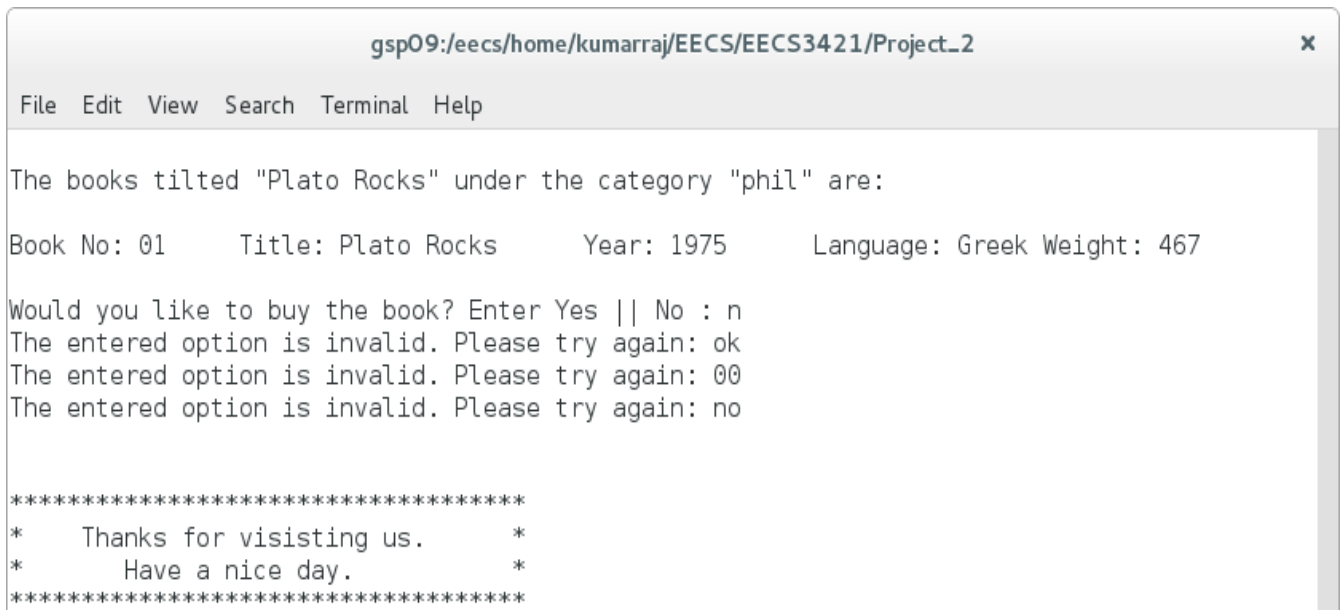
The books titled "Plato Rocks" under the category "phil" are:

Book No: 01      Title: Plato Rocks      Year: 1975      Language: Greek Weight: 467

```

If the entered book title of the chosen category exists, then we display all the details of the book. If there exists a book with same name of the same category but different year of publication or language, then all such books are displayed for the user to choose from. Each such books are displayed with a book number beside the details, making it easier for the user to choose from the books. Now we ask if the user is interested in purchasing the book with yes/no option. If the user enters no, we simply print a thanking message and we exit the program. If the user enters anything other the options provided, the program keeps prompting until a right option is entered as seen in above cases. A glimpse of the above process is shown below:

Input example for – Buying the book: String (Yes | No)



```

gsp09:/eecs/home/kumarraj/EECS/EECS3421/Project_2
File Edit View Search Terminal Help

The books tilted "Plato Rocks" under the category "phil" are:

Book No: 01      Title: Plato Rocks      Year: 1975      Language: Greek Weight: 467

Would you like to buy the book? Enter Yes || No : n
The entered option is invalid. Please try again: ok
The entered option is invalid. Please try again: 00
The entered option is invalid. Please try again: no

*****
*   Thanks for visisting us.   *
*   Have a nice day.         *
*****

```

If the user enters “yes”, then we ask the user to input the book number that he/she wishes to buy. If an invalid value or type is entered, an error message is displayed, and we continue to prompt until a valid number is entered. Then we display the minimum price for the chosen book number and ask the user to input the quantity of the book to be bought. Here again, if an invalid value or type is entered, an error message is displayed, and we continue to prompt until a valid number is entered. After that being done, we update the yrb\_purchase table with the new transaction details and display the user with the conformation message including the total price

(Quantity \* minimum price) and exits the program with a thanking message. A glimpse of the above process is shown below:

Input example for – Book number to buy: Integer (1 to n) n-number of books

Input example for – Quantity of books: Integer (1 to n) n-number of books to buy

```
The books tilted "Plato Rocks" under the category "phil" are:
Book No: 01      Title: Plato Rocks      Year: 1975      Language: Greek Weight: 467

Would you like to buy the book? Enter Yes || No : Yes

Enter the book no you wish to buy : one
Invalid Book No! Please enter a valid number to proceed: 02
Invalid Book No! Please enter a valid number to proceed: 1

The minimum price of the book "btitle" is : 36.95
Enter the number of books you wish to buy : 0
Sorry the cart seems to be empty, Please enter a valid number to proceed: ten
Sorry the cart seems to be empty, Please enter a valid number to proceed: 10

*****
*Transaction Complete!
*Customer ID : 5
*Club       : AARP
*Title      : Plato Rocks
*Year       : 1975
*Quantity   : 10
*Total Price : 369.5
*Time       : 2017-12-05-14.30.50
*Thank you, please visit us again.
*****
gsp09 91 % █
```

Whenever there is prompt to input something during the execution of the program, no matter what trash or invalid type values the user inputs related to the asked value, the program is designed in such a way that it takes in the input and reacts accordingly. This was achieved through the help of matching the user inputs to the required regular expressions in java.

## Instructions Set

Assuming this is the first time the program application is being set up or executed. We first must set up a connection to the database using the command *% db2 connect to c3421a* and use the script yrb-create, using the command *% db2 -tf yrb-create* which will create the YRB DB schema for you. It will also populate the tables with mock data. Once this is completed successfully, we do the sourcing using the command *% source ~db2leduc/cshrc.runtime*. Then we compile and execute our java program using the commands *% javac CustApp.java* and *% java CustApp* respectively. As the program is running we search through purchase the books from the database. We do this by reacting to the input asked by the program, if everything is entered properly the program will reach the end and exit. If we mess things up to the tables of the database, we can use the script yrb-drop using the command *% db2 -tf yrb-drop* and re-create our copy of YRB DB from the DB2 schema space. A glimpse of the instruction on the terminal is shown below:

Once you have a window (and thus, shell) open on a PRISM machine (e.g., red.cs.yorku.ca), go to the directory that contains the required java and yrb files and you can do the following:

Step 1: *% db2 connect to c3421a*

Step 2: *% db2 -tf yrb-create*

Step 3: *% source ~db2leduc/cshrc.runtime*

Step 4: *% javac CustApp.java*

Step 5: *% java CustApp*

```

gsp09:/eecs/home/kumarraj/EECS/EECS3421/Project_2
File Edit View Search Terminal Help
gsp09 68 % db2 connect to c3421a

Database Connection Information

Database server          = DB2/LINUX8664 11.1.1
SQL authorization ID    = KUMARRAJ
Local database alias    = C3421A

gsp09 69 % db2 -tf yrb-create

DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.

DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.

gsp09 70 % source ~/db2educ/cshrc.runtime
gsp09 71 % javac CustApp.java
gsp09 72 % java CustApp

Enter the Customer ID: 5

Customer ID: 5 Customer Name: Andy Aardverk City: Newport News

Would you like to update the customer information? Enter Yes || No : █

```