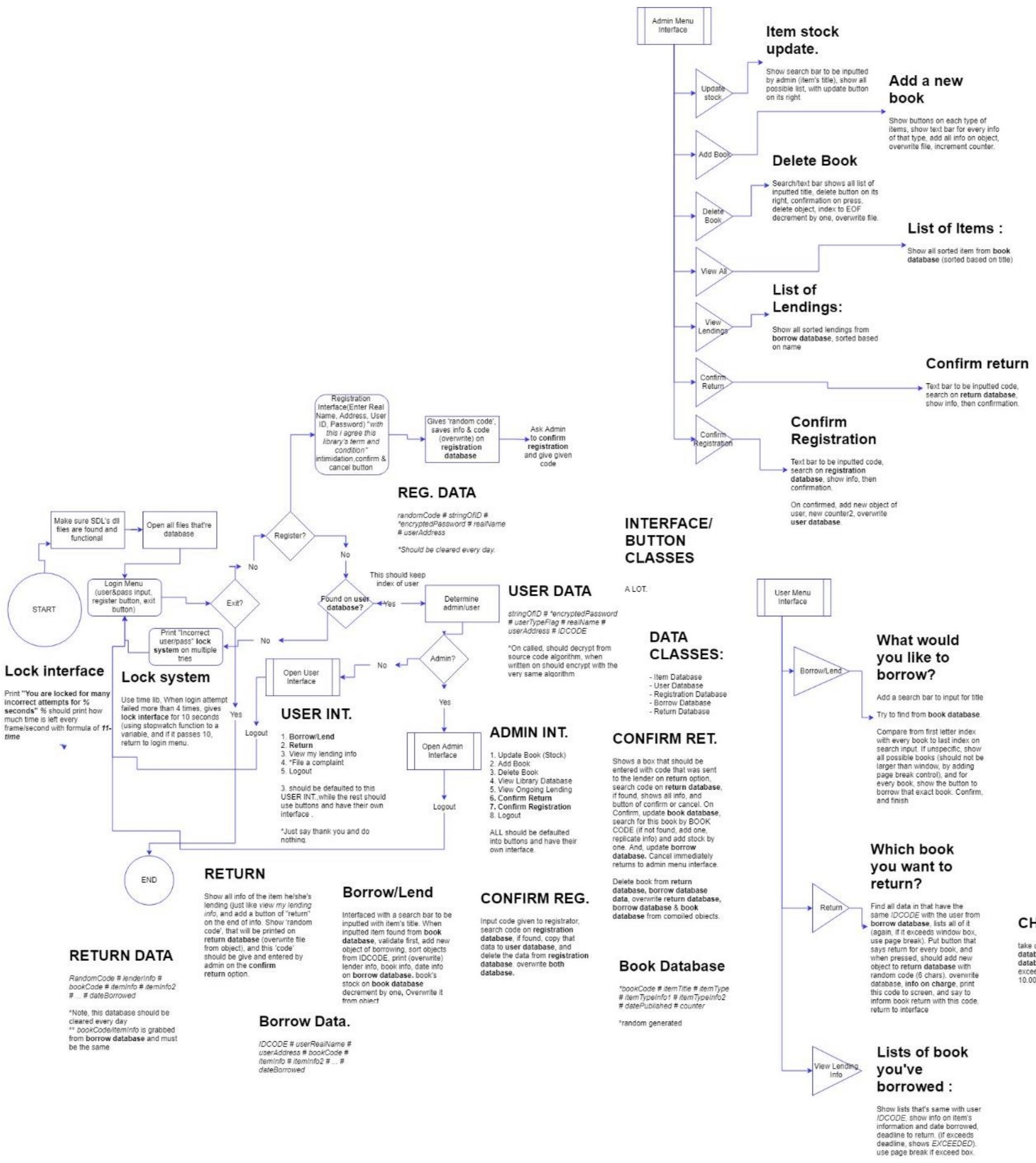
Final Term Project	BINUS UNIVERSITY
COMP6175 Object Oriented Programming	
Teknik Informatika (Master Track) *Computer Science*	**Anggota Kelompok** *Members* Jonathan Sebastian P. (2101626626) Nicholas Dominic (2101628594) Arvin Melvillo (2101628650) Hendy (2101632225)
09 Juni 2018 *June 09th, 2018*	

LIBRARY APP


START
Make sure SDL's dll files are found and functional
Open all files that're database
Login Menu (user&pass input, register button, exit button)
Exit?
No
Yes
Register?
No
Registration Interface(Enter Real Name, Address, User ID, Password) "with this i agree this library's term and condition" intimidation,confirm & cancel button
Gives 'random code', saves info & code (overwrite) on registration database
Ask Admin to confirm registration and give given code
Found on user database?
Yes
This should keep index of user
Determine admin/user
No
Print "Incorrect user/pass" lock system on multiple tries
Admin?
No
Yes
Open User Interface
Open Admin Interface
Logout
Logout
END
Admin Menu Interface
Update stock
Add Book
Delete Book
View All
View Lendings
Confirm Return
Confirm Registration
User Menu Interface
Borrow/Lend
Return
View Lending Info

Item stock update.
Show search bar to be inputted by admin (item's title), show all possible list, with update button on its right

Add a new book
Show buttons on each type of items, show text bar for every info of that type, add all info on object, overwrite file, increment counter.

Delete Book
Search/text bar shows all list of inputted title, delete button on its right, confirmation on press, delete object, index to EOF decrement by one, overwrite file.

List of Items :
Show all sorted item from **book database** (sorted based on title)

List of Lendings:
Show all sorted lendings from **borrow database**, sorted based on name

Confirm return
Text bar to be inputted code, search on **return database**, show info, then confirmation.

Confirm Registration
Text bar to be inputted code, search on **registration database**, show info, then confirmation.

On confirmed, add new object of user, new counter2, overwrite **user database**.

REG. DATA
*randomCode # stringOfID # *encryptedPassword # realName # userAddress*

**Should be cleared every day.*

INTERFACE/ BUTTON CLASSES
A LOT.

USER DATA
*stringOfID # *encryptedPassword # userTypeFlag # realName # userAddress # IDCODE*

*On called, should decrypt from source code algorithm, when written on should encrypt with the very same algorithm

DATA CLASSES:
- Item Database
- User Database
- Registration Database
- Borrow Database
- Return Database

Lock interface
Print **"You are locked for many incorrect attempts for % seconds"** % should print how much time is left every frame/second with formula of ***11-time***

Lock system
Use time lib, When login attempt failed more than 4 times, gives **lock interface** for 10 seconds (using stopwatch function to a variable, and if it passes 10, return to login menu.

USER INT.
1. **Borrow/Lend**
2. **Return**
3. View my lending info
4. *File a complaint
5. Logout

3. should be defaulted to this USER INT.,while the rest should use buttons and have their own interface .

*Just say thank you and do nothing.

ADMIN INT.
1. Update Book (Stock)
2. Add Book
3. Delete Book
4. View Library Database
5. View Ongoing Lending
6. **Confirm Return**
7. **Confirm Registration**
8. Logout

ALL should be defaulted into buttons and have their own interface.

CONFIRM RET.
Shows a box that should be entered with code that was sent to the lender on **return** option, search code on **return database**, if found, shows all info, and button of confirm or cancel. On Confirm, update **book database**, search for this book by BOOK CODE (if not found, add one, replicate info) and add stock by one. And, update **borrow database.** Cancel immediately returns to admin menu interface.

Delete book from **return database, borrow database data**, overwrite **return database, borrow database** & **book database** from compiled objects.

What would you like to borrow?
Add a search bar to input for title

Try to find from **book database**.

Compare from first letter index with every book to last index on search input. If unspecific, show all possible books (should not be larger than window, by adding page break control), and for every book, show the button to borrow that exact book. Confirm, and finish

RETURN
Show all info of the item he/she's lending (just like *view my lending info*, and add a button of "return" on the end of info. Show 'random code', that will be printed on **return database** (overwrite file from object), and this 'code' should be give and entered by admin on the **confirm return** option.

Borrow/Lend
Interfaced with a search bar to be inputted with item's title. When inputted item found from **book database**, validate first, add new object of borrowing, sort objects from IDCODE, print (overwrite) lender info, book info, date info on **borrow database.** book's stock on **book database** decrement by one**,** Overwrite it from object

CONFIRM REG.
Input code given to registrator, search code on **registration database**, if found, copy that data to **user database**, and delete the data from **registration database**. overwrite **both database.**

RETURN DATA
RandomCode # lenderInfo # bookCode # itemInfo # itemInfo2 # ... # dateBorrowed

*Note, this database should be cleared every day
** *bookCode/itemInfo* is grabbed from **borrow database** and must be the same

Borrow Data.
IDCODE # userRealName # userAddress # bookCode # itemInfo # itemInfo2 # ... # dateBorrowed

Book Database
**bookCode # itemTitle # itemType # itemTypeInfo1 # itemTypeInfo2 # datePublished # counter*

*random generated

Which book you want to return?
Find all data in that have the same *IDCODE* with the user from **borrow database**, lists all of it (again, if it exceeds window box, use page break). Put button that says return for every book, and when pressed, should add new object to **return database** with random code (6 chars). overwrite database, **info on charge**, print this code to screen, and say to inform book return with this code. return to interface

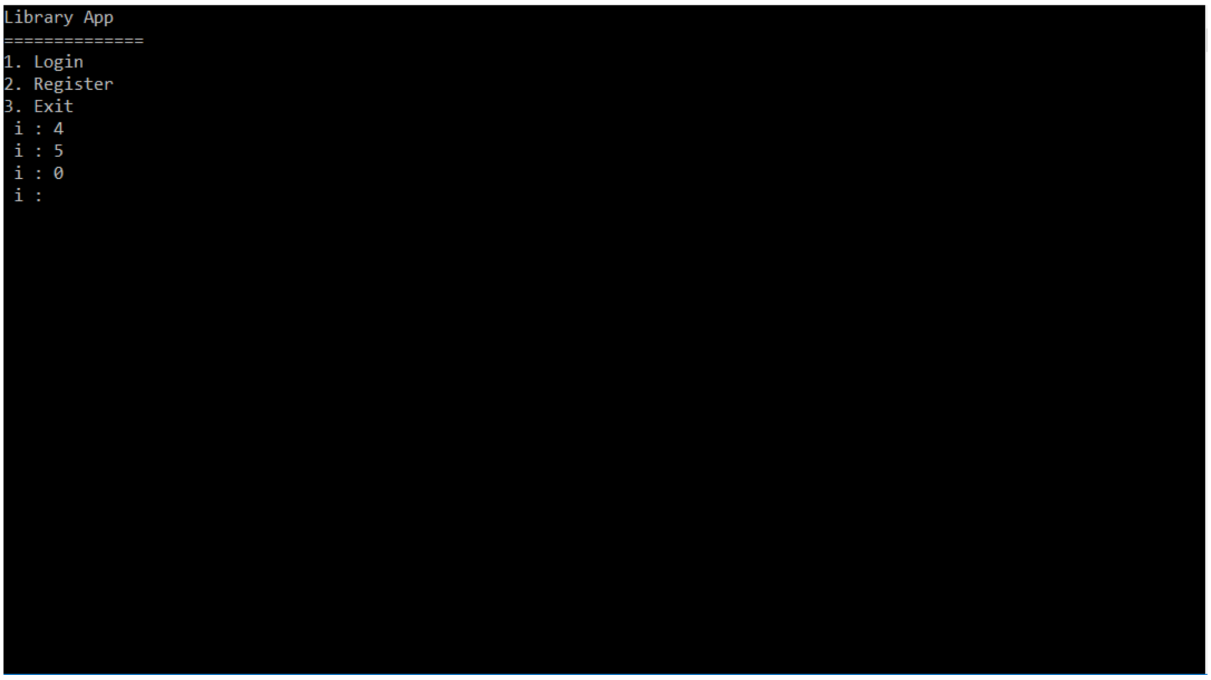
CHARGE INFO
take *dateBorrowed* from **return database**, taken from **borrow database**, then see date, if it exceeds 14 days, charge Rp. 10.000 per day.

Lists of book you've borrowed :
Show lists that's same with user *IDCODE*, show info on item's information and date borrowed, deadline to return. (if exceeds deadline, shows *EXCEEDED*). use page break if exceed box.

** activities that're outside the program

The library app is a program to help users (librarians/ readers) on processing datas, such as, see all listed available items, lending an item & returning an item with keeping trace of user, updating item's stock, etc.
The program is command-line interfaced, needing input from its user to run its program.

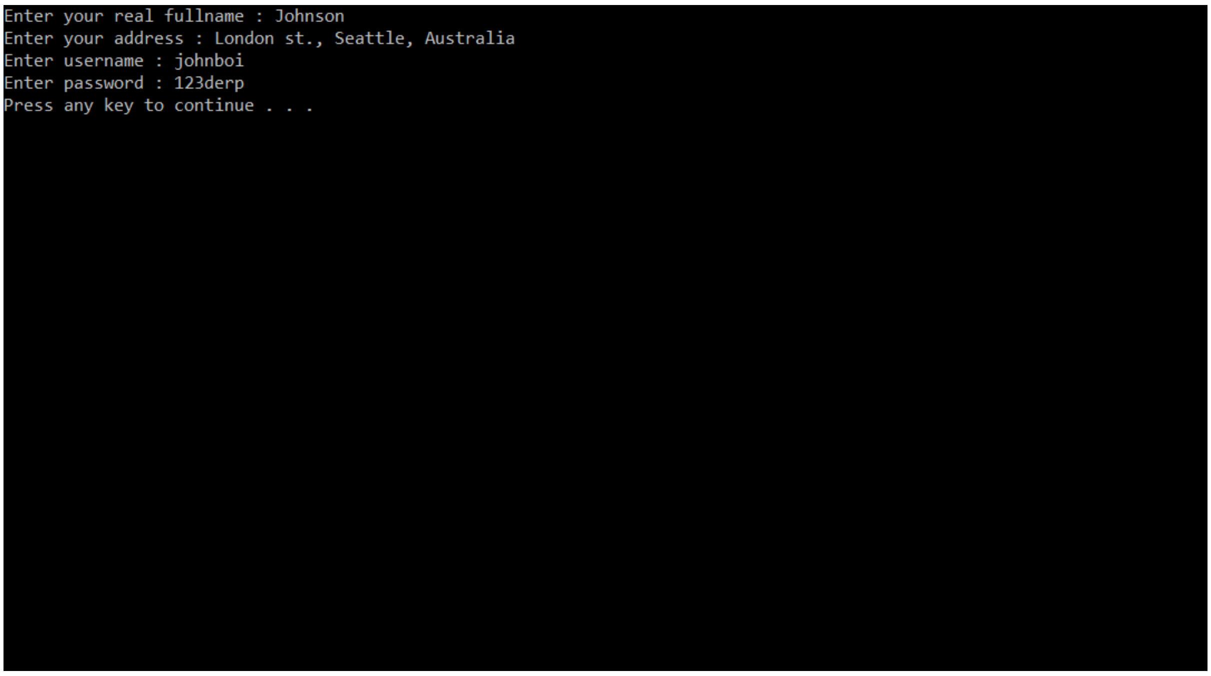
LOGIN/ REGISTER MAIN MENU



```
Library App
==============
1. Login
2. Register
3. Exit
 i : 4
 i : 5
 i : 0
 i :
```

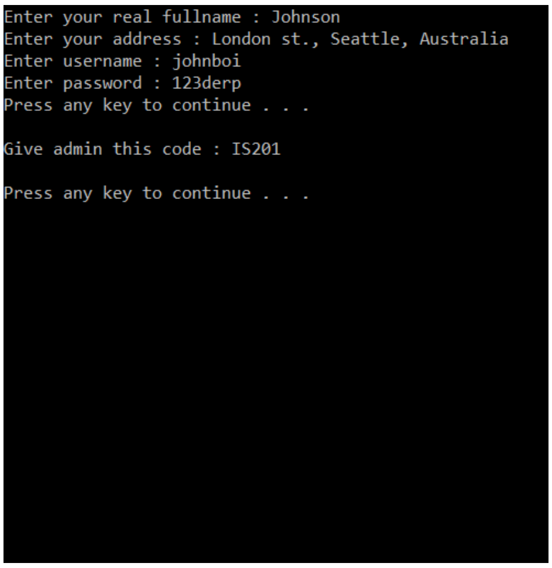
On opening the app, the login/ register menu will appear, asking either if user wants to login, register, or exit application. User input will be validated to make sure its input is within the program options.

REGISTER MENU

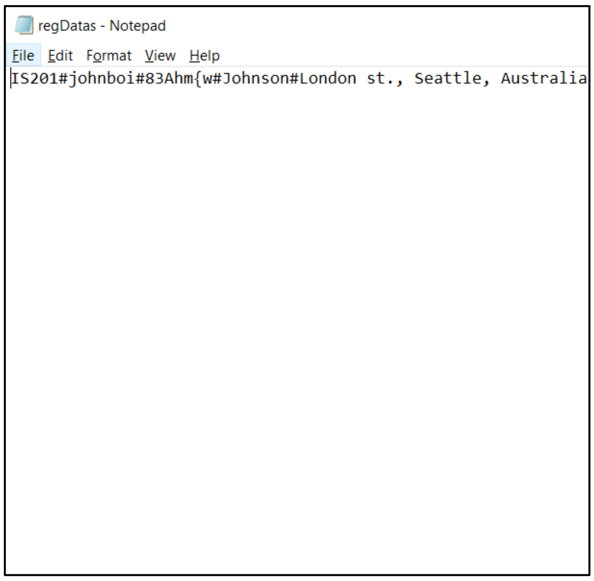


```
Enter your real fullname : Johnson
Enter your address : London st., Seattle, Australia
Enter username : johnboi
Enter password : 123derp
Press any key to continue . . .
```

When opening the register menu, user will be asked to input his/her name, address, username, and password. After they enter all datas, they are **NOT** registered as a user yet.



```
Enter your real fullname : Johnson
Enter your address : London st., Seattle, Australia
Enter username : johnboi
Enter password : 123derp
Press any key to continue . . .

Give admin this code : IS201

Press any key to continue . . .
```

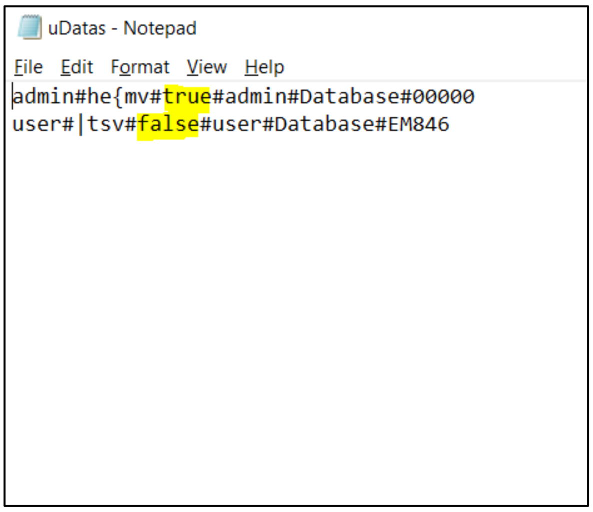


```
IS201#johnboi#83Ahm{w#Johnson#London st., Seattle, Australia
```

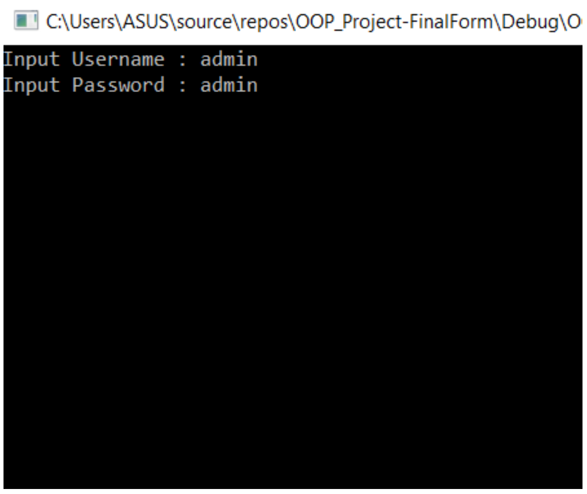
After they finish entered all data needed, they are **NOT** registered as a library user yet. Instead, they will be shown a random code , at which must be given to the librarian** so the data can be validated by the librarian from the registree identity and the data inserted on the database to confirm registration. The data is saved on the **Register Database**.

After that, they will be returned to the **Login/ Register Main Menu.**

LOGIN MENU

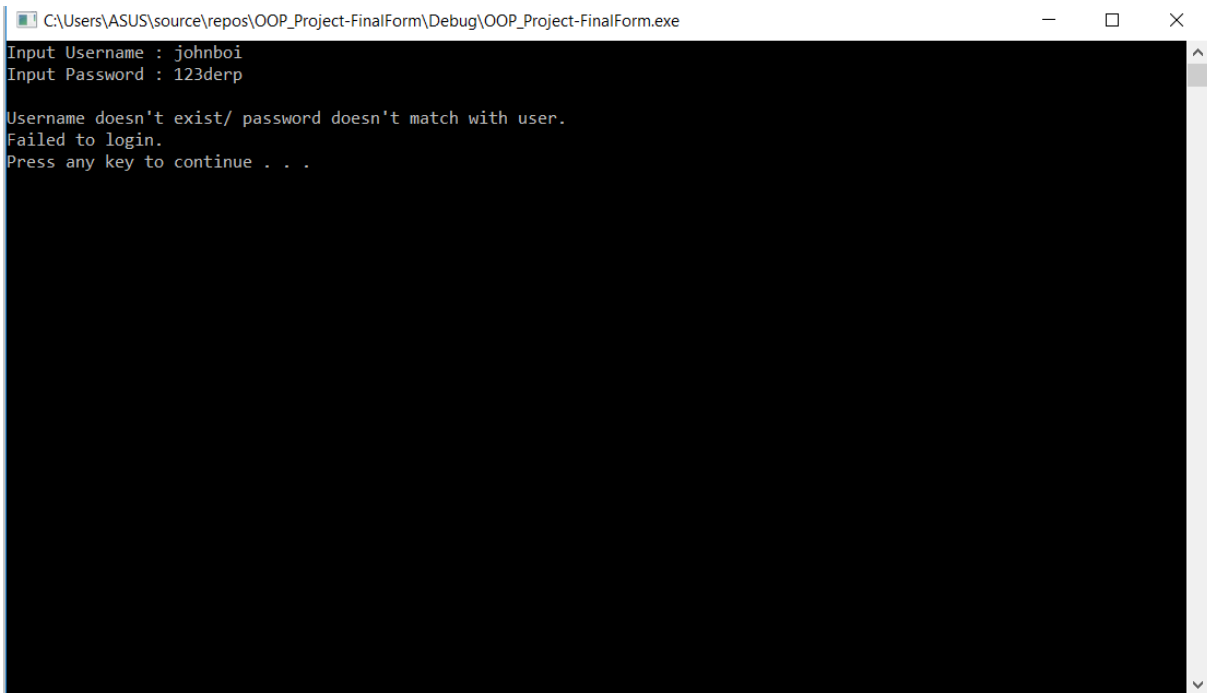


```
admin#he{mv#true#admin#Database#00000
user#|tsv#false#user#Database#EM846
```



```
Input Username : admin
Input Password : admin
```

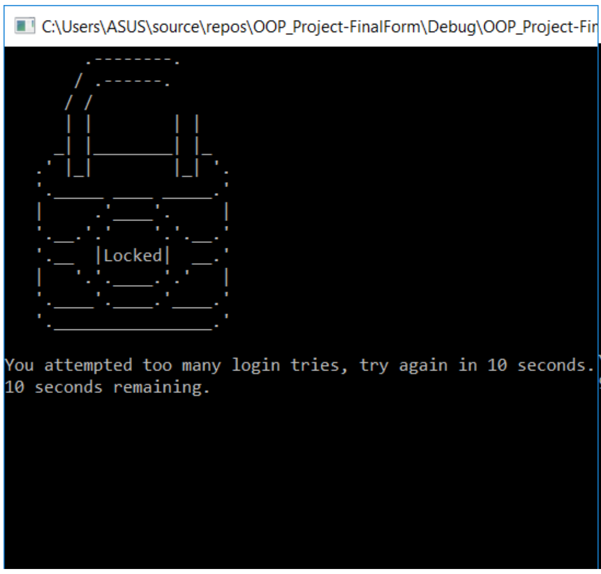
When opening login menu, the program will open the **User Database** and read all of the data and put each line as a class object. Each line in the database contain username, encrypted password, user type (highlighted), user real name, user address, and user ID code. Then, the program will ask the user to input his/her username and password, then match the username and decypted password with the data.


C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-FinalForm.exe

```
Input Username : johnboi
Input Password : 123derp

Username doesn't exist/ password doesn't match with user.
Failed to login.
Press any key to continue . . .
```

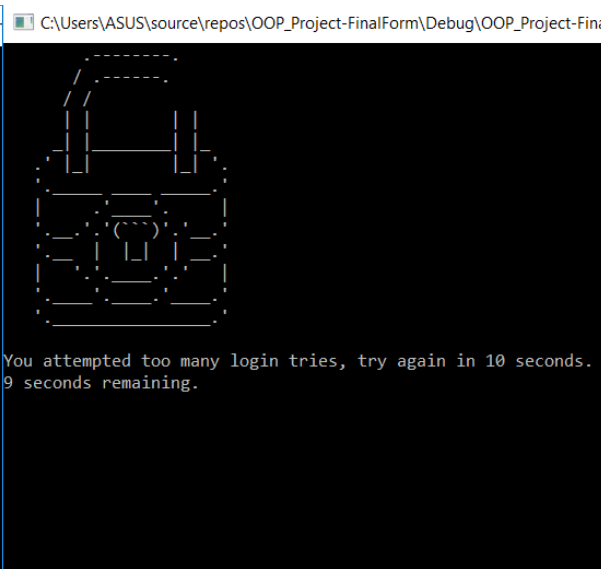
If the program fails to get a match from any of the user data, the program will return to **Login/ Register Main Menu**. However, keeping counter on how many times has the user failed to login. If the user attempted too much login without success, the program will run **Lock System**.

LOCK SYSTEM



C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-Fir

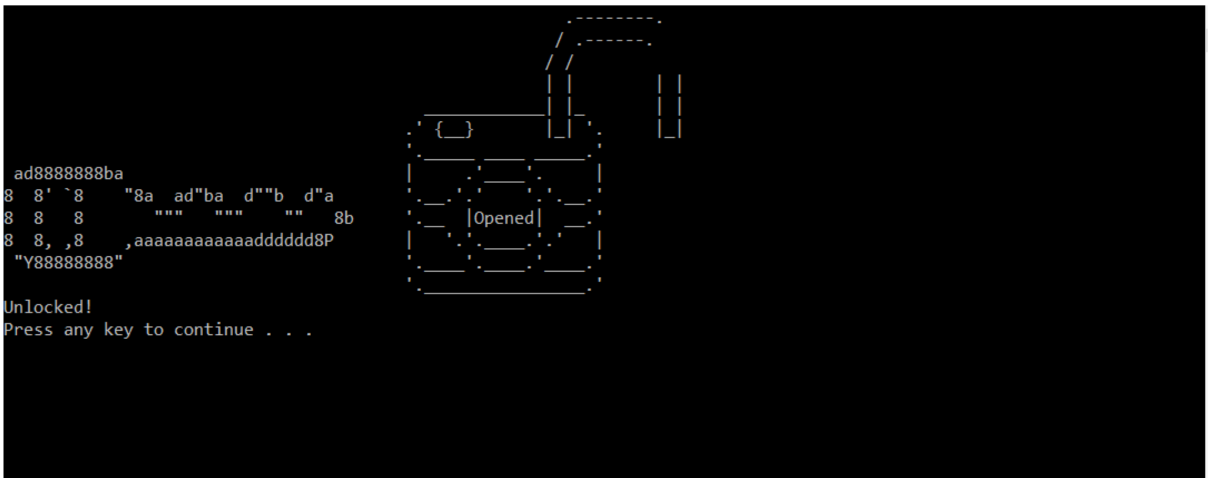
```
         .--------.
        / .------.
       / /
       | |        | |
      _| |________| |_
    .' |_|        |_| '.
    '._____ ____ _____.'
    |     .'____'.     |
    '.__.'.'    '.'.__.'
    '.__  |Locked|  __.'
    |   '.'.____.'.'   |
    '.____'.____.'____.'
    '.________________.'

You attempted too many login tries, try again in 10 seconds.
10 seconds remaining.
```



C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-Fina

```
         .--------.
        / .------.
       / /
       | |        | |
      _| |________| |_
    .' |_|        |_| '.
    '._____ ____ _____.'
    |     .'____'.     |
    '.__.'.(```).'.__.'
    '.__  | |_| |  __.'
    |   '.'.____.'.'   |
    '.____'.____.'____.'
    '.________________.'

You attempted too many login tries, try again in 10 seconds.
9 seconds remaining.
```

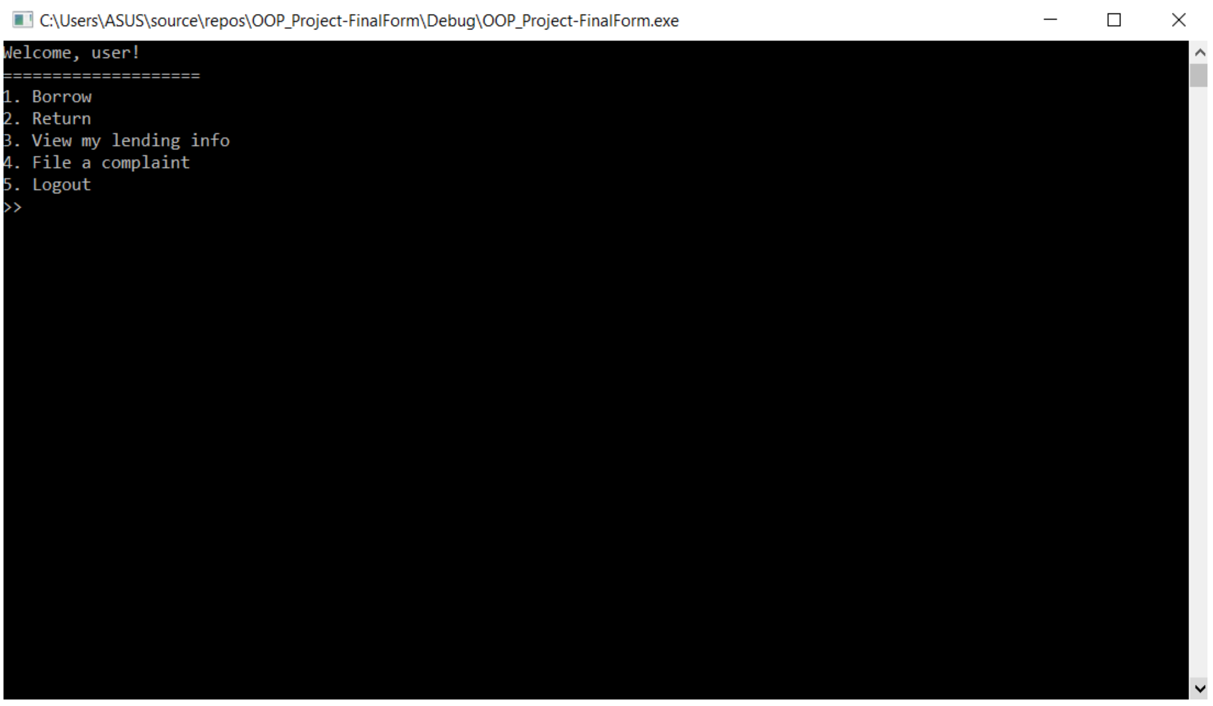
If the user tried 5 login attempts without success, the program will lock itself down, while displaying a timer and a message. The user must wait until the program timer ran out. Then, the program will return to **Login/ Register Main Menu.**



```
                                                        .--------.
                                                       / .------.
                                                      / /
                                                      | |        | |
                                          ____________| |_       | |
                                        .' {__}       |_| '.     |_|
                                        '                  '
                                        '._____ ____ _____.'
 ad8888888ba                            |     .'____'.     |
8  8' `8    "8a  ad"ba  d""b  d"a       '.__.'.'    '.'.__.'
8  8   8       """   """    ""   8b     '.__  |Opened|  __.'
8  8, ,8    ,aaaaaaaaaaaadddddd8P       |   '.'.____.'.'   |
 "Y88888888"                            '.____'.____.'____.'
                                        '.________________.'
Unlocked!
Press any key to continue . . .
```

If the program found a match, however, the program will copy that user data as a new object as known as *This User* (to imply who is running the function, and easier access to save that user data when needed) and the program can do one of two things, open **Admin Interface Menu** or open **User Interface Menu**. The way the program know which interface menu it should open is by reading its user type in boolen form from database wether it's an admin (true) or a user (false). the false value will open **User Interface Menu**, while true value will open the other.

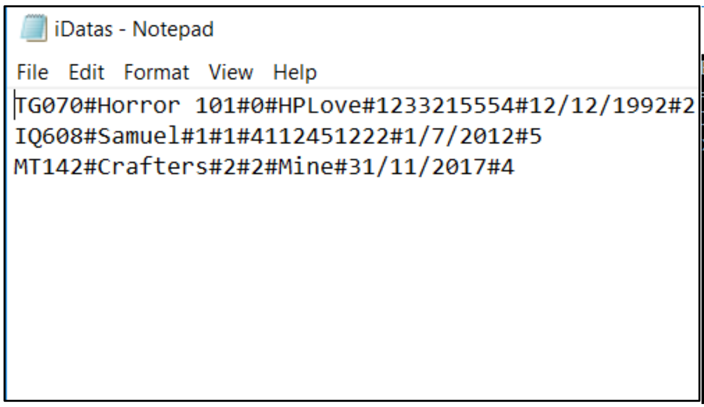
USER INTERFACE MENU


C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-FinalForm.exe

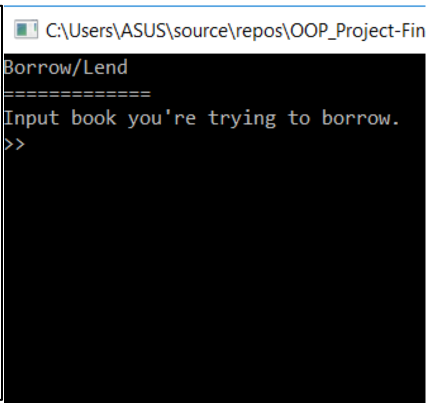
```
Welcome, user!
====================
1. Borrow
2. Return
3. View my lending info
4. File a complaint
5. Logout
>>
```

This menu consists of 5 options, **Borrow Option**, **Return Option**, **View My Lending**, **File Complaint Option**, Logout option. Logout simply returns to **Login/ Register Main** Menu. User will be asked to choose within these options. input will be validated.

BORROW OPTION

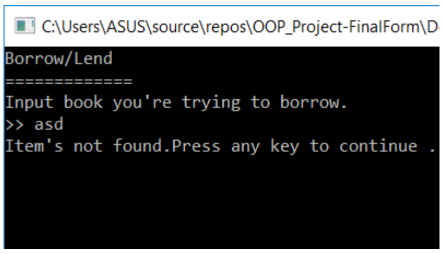

iDatas - Notepad

```
TG070#Horror 101#0#HPLove#1233215554#12/12/1992#2
IQ608#Samuel#1#1#4112451222#1/7/2012#5
MT142#Crafters#2#2#Mine#31/11/2017#4
```

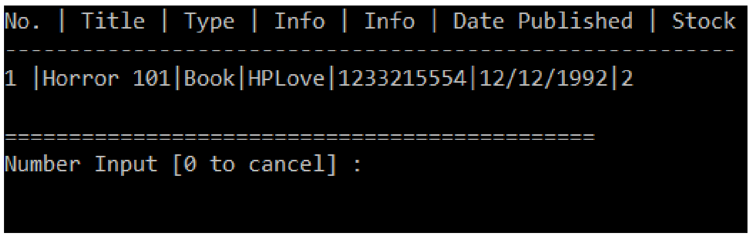

C:\Users\ASUS\source\repos\OOP_Project-Fin

```
Borrow/Lend
============
Input book you're trying to borrow.
>>
```

Upon opening the borrow option, program will read all data on the **Item Database**. Then user will be asked to input the item's title, which then the program will try to find. The way how the program search is by **Search Each Word**.

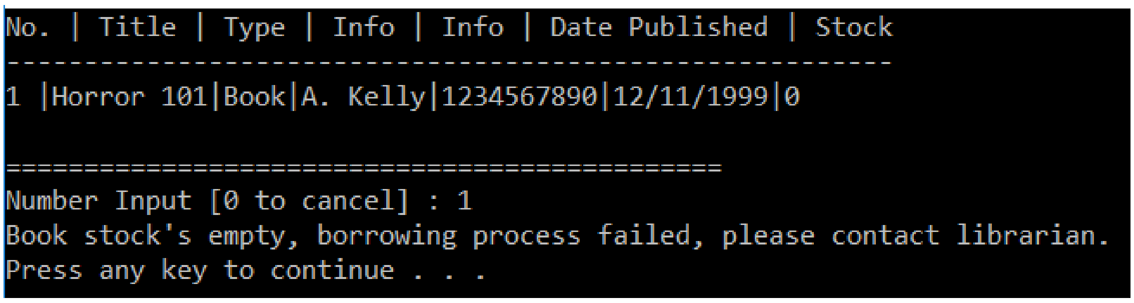

C:\Users\ASUS\source\repos\OOP_Project-FinalForm\D

```
Borrow/Lend
============
Input book you're trying to borrow.
>> asd
Item's not found.Press any key to continue .
```



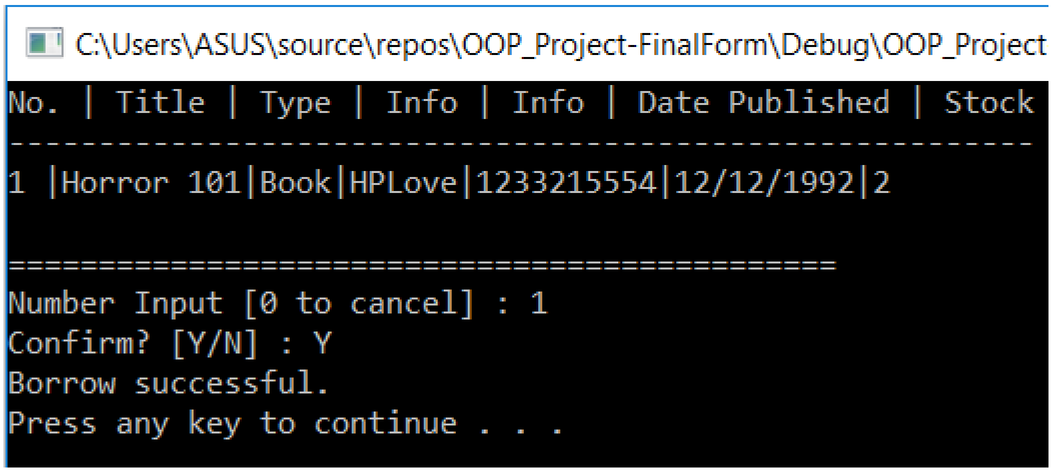
```
No. | Title | Type | Info | Info | Date Published | Stock
-----------------------------------------------------------
1  |Horror 101|Book|HPLove|1233215554|12/12/1992|2

==============================================
Number Input [0 to cancel] :
```

If there's no item found, program showed that there is no found item. Else, show every item that match input, and ask input on which item to borrow.

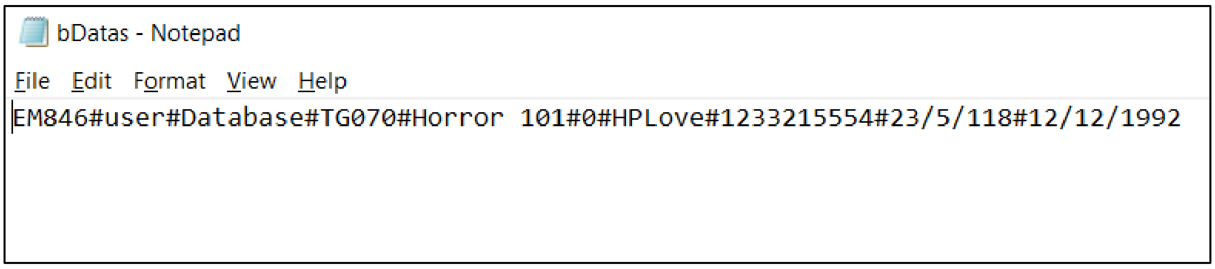


```
No. | Title | Type | Info | Info | Date Published | Stock
-----------------------------------------------------------
1  |Horror 101|Book|A. Kelly|1234567890|12/11/1999|0

==============================================
Number Input [0 to cancel] : 1
Book stock's empty, borrowing process failed, please contact librarian.
Press any key to continue . . .
```

After input on an item, the program will see if the stock is not empty. if its' empty, the program will display the stock is empty.



```
C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project
No. | Title | Type | Info | Info | Date Published | Stock
--------------------------------------------------------------
1 |Horror 101|Book|HPLove|1233215554|12/12/1992|2


=============================================
Number Input [0 to cancel] : 1
Confirm? [Y/N] : Y
Borrow successful.
Press any key to continue . . .
```



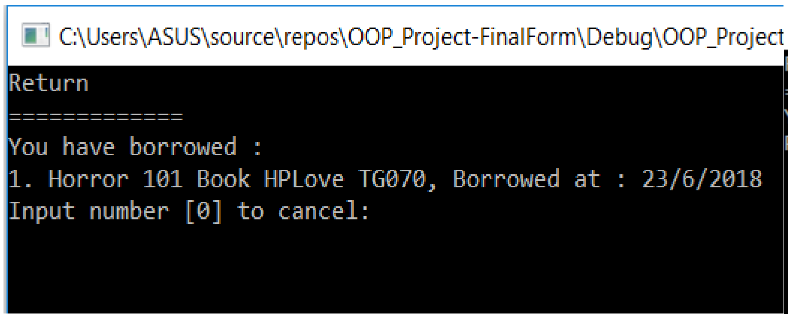
```
bDatas - Notepad
File  Edit  Format  View  Help
EM846#user#Database#TG070#Horror 101#0#HPLove#1233215554#23/5/118#12/12/1992
```

If it's not empty, it'll proceed to ask the user for confirmation, and when they do, the program will save *This User* info (ID code, name, address) info and the Item info (ID Code, title, info, other info, item published date, and borrowed date) on the **Borrowed Database**.

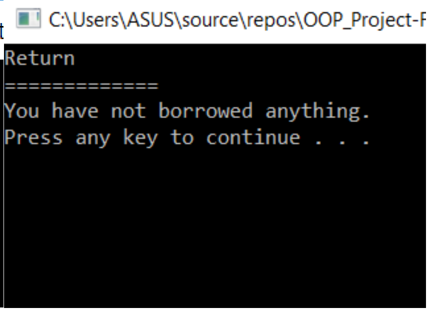
SEARCH EACH WORD

Searching function, where user input's string will be compared with each word in each item. the program will try to compare each letter from the first letter of the user input and the first letter of the first item's word (before a space) when comparing each letter, if a letter is not the same, it'll flag it as not the item user inputted, and then search again on the second's word of the title(after a space) and compare it with the input (until there's no more space). This'll loop until all item has been compared.

RETURN OPTION

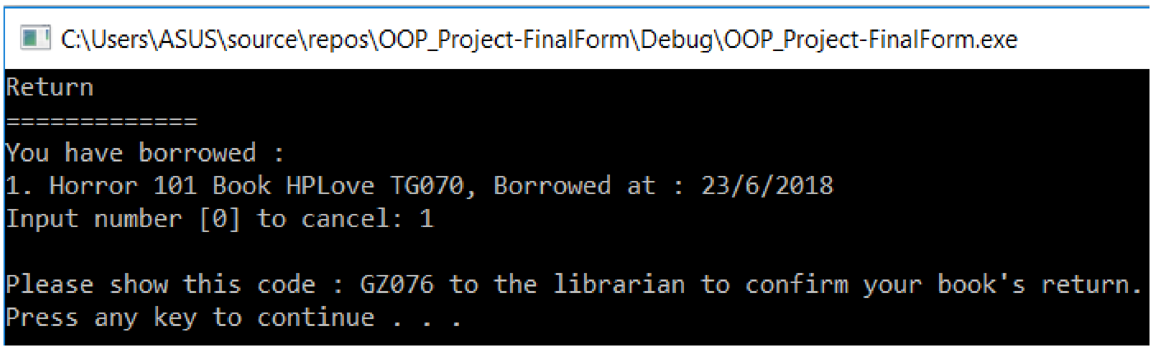


```
C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project
Return
=============
You have borrowed :
1. Horror 101 Book HPLove TG070, Borrowed at : 23/6/2018
Input number [0] to cancel:
```

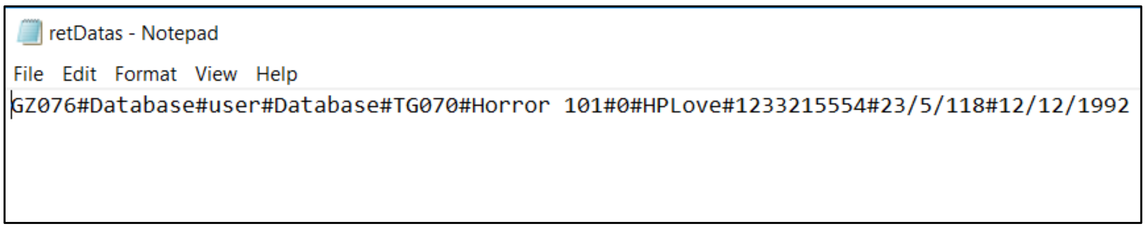


```
C:\Users\ASUS\source\repos\OOP_Project-F
Return
=============
You have not borrowed anything.
Press any key to continue . . .
```

When opening this option, the program will read every data from **Borrowed Database**, and compare *This User* ID code with each item's borrower ID code. Then it'll show every item that's borrowed by *This User* (none if user doesn't borrow anything). It'll then ask input on the index number on which item the user would like to return.



```
C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-FinalForm.exe
Return
=============
You have borrowed :
1. Horror 101 Book HPLove TG070, Borrowed at : 23/6/2018
Input number [0] to cancel: 1

Please show this code : GZ076 to the librarian to confirm your book's return.
Press any key to continue . . .
```

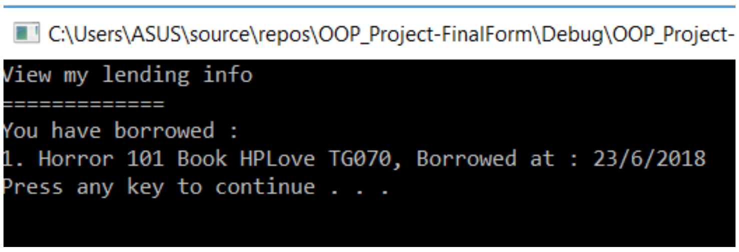


```
retDatas - Notepad
File  Edit  Format  View  Help
GZ076#Database#user#Database#TG070#Horror 101#0#HPLove#1233215554#23/5/118#12/12/1992
```

When entering a item the user would like to return, the user will be shown a random code that must be entered by the librarian to confirm it. This function does **NOT** mean the user has returned the item.

Data from the **Borrowed Database** will be copied to the **Return Database**.

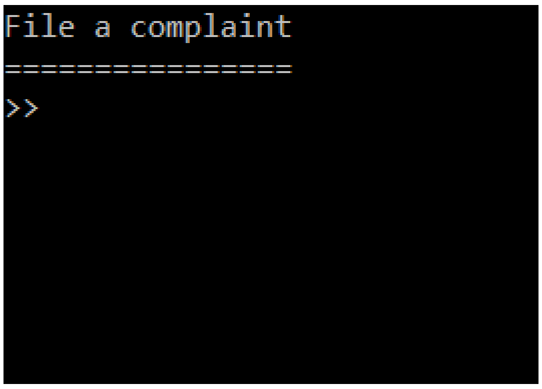
VIEW MY LENDING



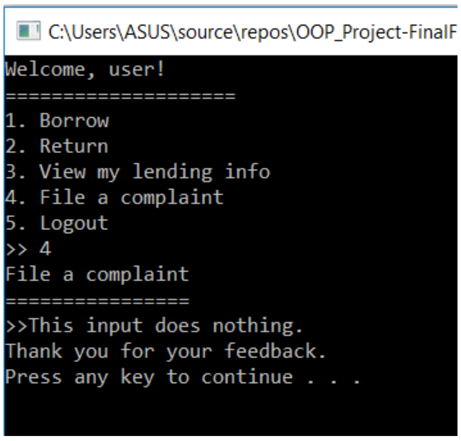
```
C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-
View my lending info
=============
You have borrowed :
1. Horror 101 Book HPLove TG070, Borrowed at : 23/6/2018
Press any key to continue . . .
```

On this option, it'll only show on every item which the user has borrowed, just like the return function.

FILE A COMPLAINT



```
File a complaint
================
>>
```

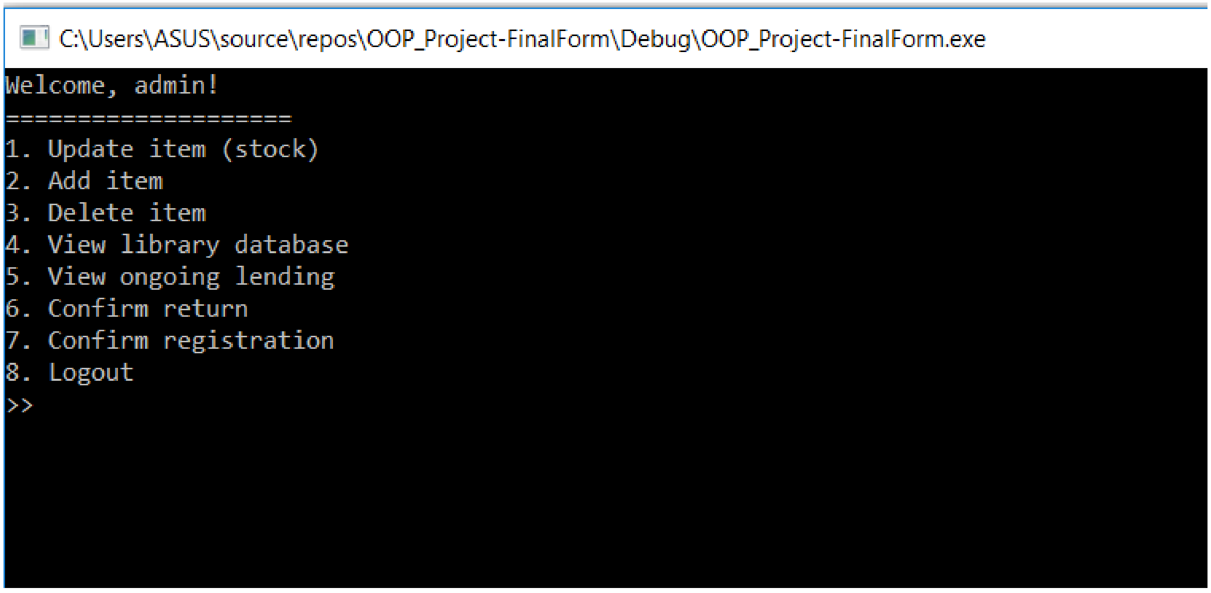


C:\Users\ASUS\source\repos\OOP_Project-FinalF

```
Welcome, user!
====================
1. Borrow
2. Return
3. View my lending info
4. File a complaint
5. Logout
>> 4
File a complaint
================
>>This input does nothing.
Thank you for your feedback.
Press any key to continue . . .
```

Is the user not satisfied with our very hard work of our services to them? That's okay, we created an option specifically for the user, to vent on a machine rather than on somebody else. Sometimes user arrogance and pride can place themselves high enough where they feel that they can do annoying actions which are rich in toxicity, that it will affects other people that're around him, such as, a librarian. That's okay, we can understand that. That's why, we created a solution, this option will ask for a user input. After input, the program will display a very polite, nice, and sweet thank you for the user. The input is not saved.

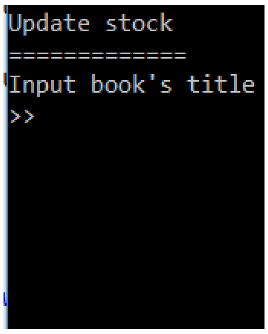
ADMIN INTERFACE MENU



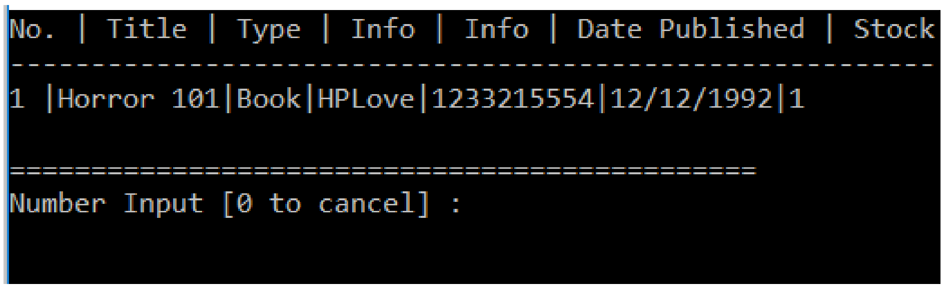
```
C:\Users\ASUS\source\repos\OOP_Project-FinalForm\Debug\OOP_Project-FinalForm.exe
Welcome, admin!
====================
1. Update item (stock)
2. Add item
3. Delete item
4. View library database
5. View ongoing lending
6. Confirm return
7. Confirm registration
8. Logout
>>
```

If the program see that the user is a admin (librarian), it'll open this option and showed these options : **Update Stock**, **Add Item**, **Delete Item**, **View All Item**, **View All Lendings**, **Confirm Return**, **Confirm Registration**¸ and Logout Option. As usual, logout option will return user to **Login/ Register Main Menu**. Input is validated.

UPDATE STOCK

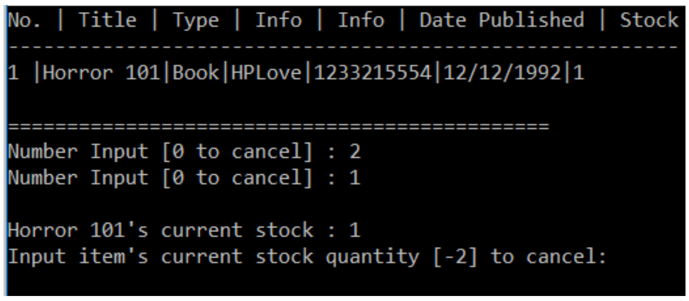


```
Update stock
============
Input book's title
>>
```

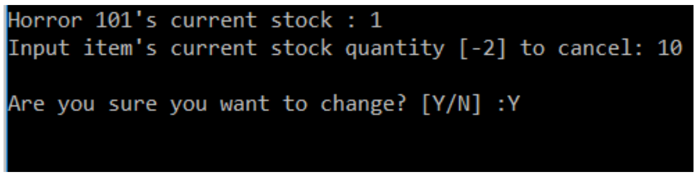


```
No. | Title | Type | Info | Info | Date Published | Stock
----------------------------------------------------------
1  |Horror 101|Book|HPLove|1233215554|12/12/1992|1

============================================
Number Input [0 to cancel] :
```

On updating stock option, the program will read every item on **Item Database** and ask for a user input. it'll search with the **Search Each Word** function, and when found, it'll display all available items.



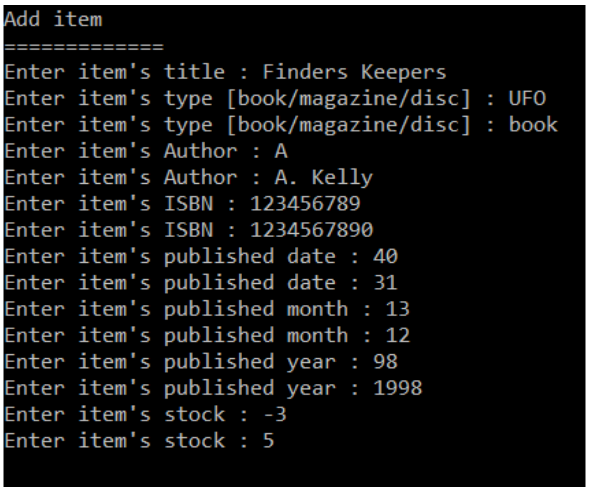
```
No. | Title | Type | Info | Info | Date Published | Stock
----------------------------------------------------------
1  |Horror 101|Book|HPLove|1233215554|12/12/1992|1

============================================
Number Input [0 to cancel] : 2
Number Input [0 to cancel] : 1

Horror 101's current stock : 1
Input item's current stock quantity [-2] to cancel:
```



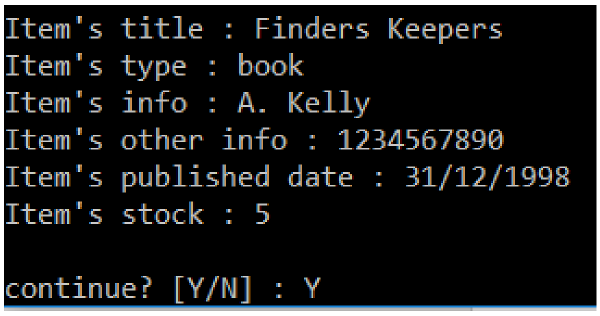
```
Horror 101's current stock : 1
Input item's current stock quantity [-2] to cancel: 10

Are you sure you want to change? [Y/N] :Y
```

After choosing a item, it'll display item's current stock, then ask for the change of stock to user. After confirmation, the program will overwrite data on **Item Database**.

ADD ITEM

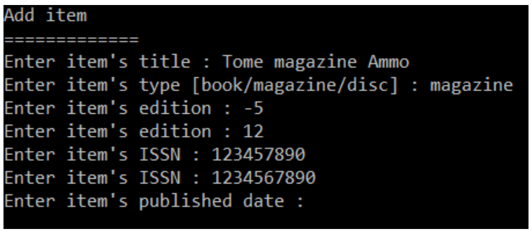


```
Add item
=============
Enter item's title : Finders Keepers
Enter item's type [book/magazine/disc] : UFO
Enter item's type [book/magazine/disc] : book
Enter item's Author : A
Enter item's Author : A. Kelly
Enter item's ISBN : 123456789
Enter item's ISBN : 1234567890
Enter item's published date : 40
Enter item's published date : 31
Enter item's published month : 13
Enter item's published month : 12
Enter item's published year : 98
Enter item's published year : 1998
Enter item's stock : -3
Enter item's stock : 5
```

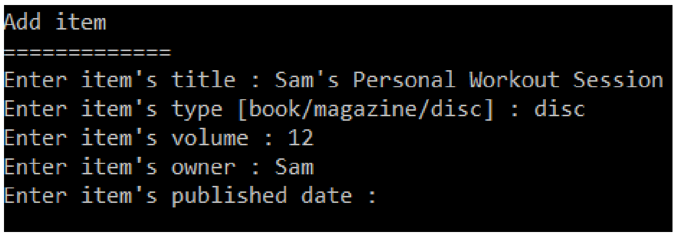


```
Item's title : Finders Keepers
Item's type : book
Item's info : A. Kelly
Item's other info : 1234567890
Item's published date : 31/12/1998
Item's stock : 5

continue? [Y/N] : Y
```

On adding an item option, it'll ask for a title, type, info, other info, published date, and stock. All of which are validated. Author needs minimum input of 3 letters, ISBN info must have 10 letters exact, date must be in calendars format, and stock can't have negative value.



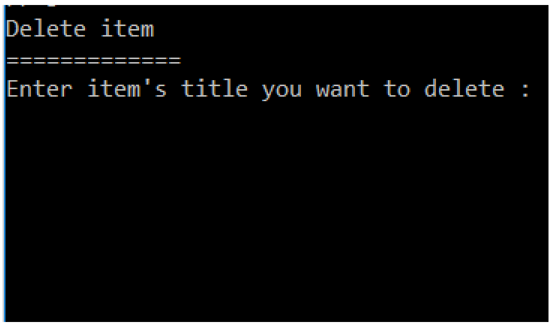
```
Add item
============
Enter item's title : Tome magazine Ammo
Enter item's type [book/magazine/disc] : magazine
Enter item's edition : -5
Enter item's edition : 12
Enter item's ISSN : 123457890
Enter item's ISSN : 1234567890
Enter item's published date :
```



```
Add item
=============
Enter item's title : Sam's Personal Workout Session
Enter item's type [book/magazine/disc] : disc
Enter item's volume : 12
Enter item's owner : Sam
Enter item's published date :
```

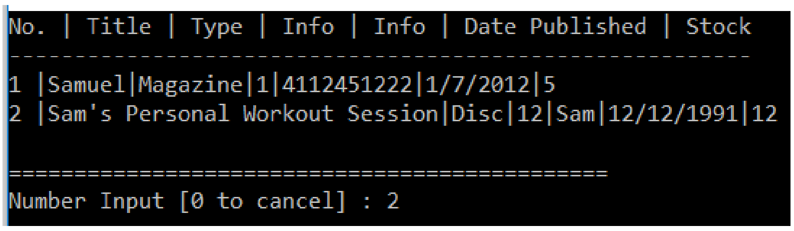
For every type, however these info are not the same (disc has volume and owner, magazine has edition and ISSN), since the program define type as different classes which inherited to another class as a same object (these parents only define the types, while the inherited class define other info, such as title, published dates, etc, to avoid diamonds problem).
After confirming on adding an item, it'll add the item to the **Item Database**.

DELETE ITEM



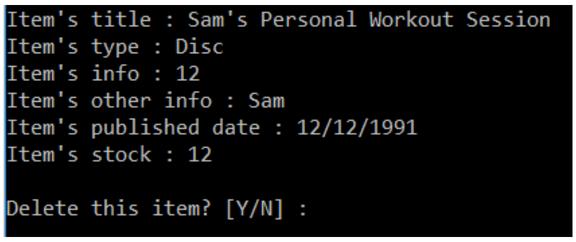
```
Delete item
=============
Enter item's title you want to delete :
```

On this option, the program will read all data from the **Item** Database. The user will be ask for input. This input will be search with **Search Each Word** function to find all available item from input.



```
No. | Title | Type | Info | Info | Date Published | Stock
----------------------------------------------------------
1 |Samuel|Magazine|1|4112451222|1/7/2012|5
2 |Sam's Personal Workout Session|Disc|12|Sam|12/12/1991|12

============================================
Number Input [0 to cancel] : 2
```

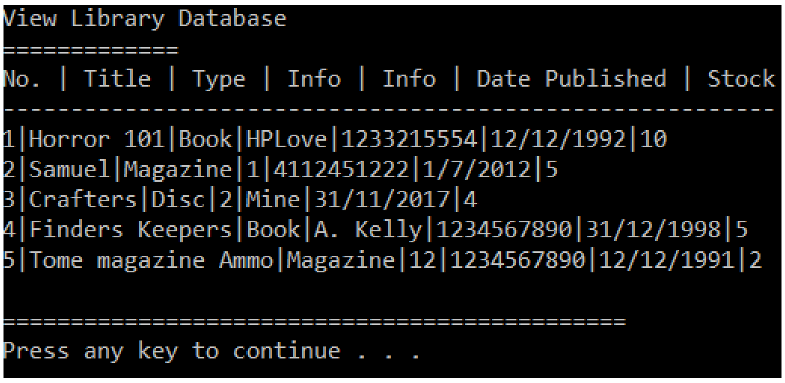
Same as usual, it'll display all item based on input, and ask for more input on the item user wants to delete.



```
Item's title : Sam's Personal Workout Session
Item's type : Disc
Item's info : 12
Item's other info : Sam
Item's published date : 12/12/1991
Item's stock : 12

Delete this item? [Y/N] :
```

After entering a item's number, it'll display all its information and ask for a confirmation. On confirm, it'll delete that specific object, overwrite all data to **Item's Database**.

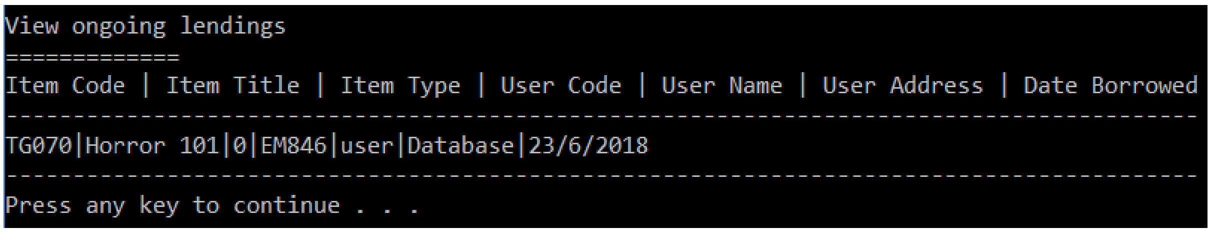
VIEW ALL ITEM



```
View Library Database
=============
No. | Title | Type | Info | Info | Date Published | Stock
---------------------------------------------------------
1|Horror 101|Book|HPLove|1233215554|12/12/1992|10
2|Samuel|Magazine|1|4112451222|1/7/2012|5
3|Crafters|Disc|2|Mine|31/11/2017|4
4|Finders Keepers|Book|A. Kelly|1234567890|31/12/1998|5
5|Tome magazine Ammo|Magazine|12|1234567890|12/12/1991|2

==============================================
Press any key to continue . . .
```

View all item simply display all items in the **Item Database**, and show all its information.

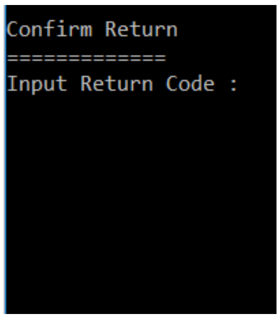
VIEW ALL LENDINGS



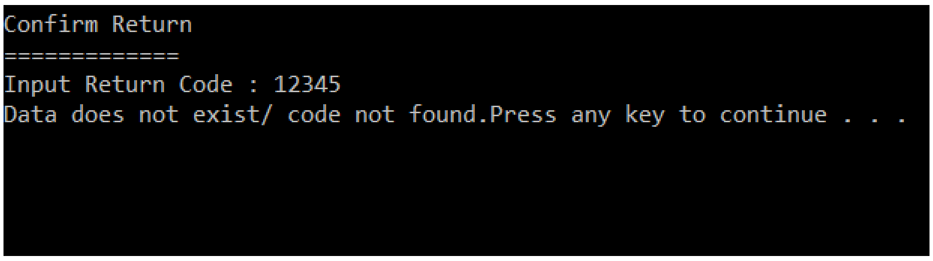
```
View ongoing lendings
=============
Item Code | Item Title | Item Type | User Code | User Name | User Address | Date Borrowed
----------------------------------------------------------------------------------------
TG070|Horror 101|0|EM846|user|Database|23/6/2018
----------------------------------------------------------------------------------------
Press any key to continue . . .
```

On view all ongoing lendings option, it’ll show all datas on **Borrowed Database**, showing all infomation from borrowers to the items.

CONFIRM RETURN

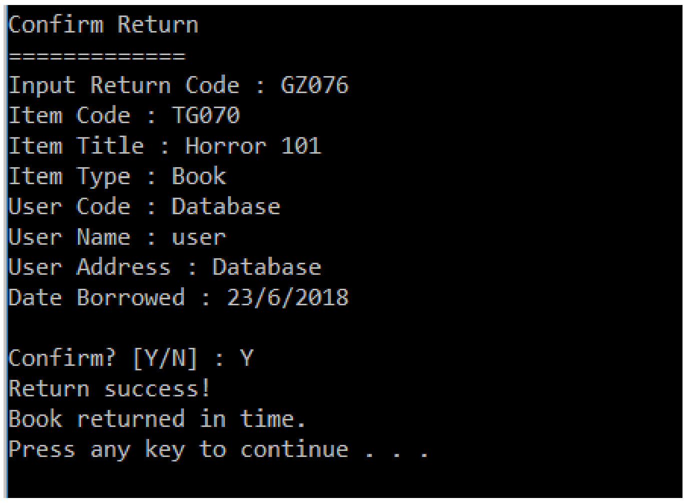


```
Confirm Return
============
Input Return Code :
```



```
Confirm Return
=============
Input Return Code : 12345
Data does not exist/ code not found.Press any key to continue . . .
```

On confirm return option, it’ll read all datas on the **Return Database**, and ask for a user input. The option will return the user to **Admin Interface Menu** when not found.

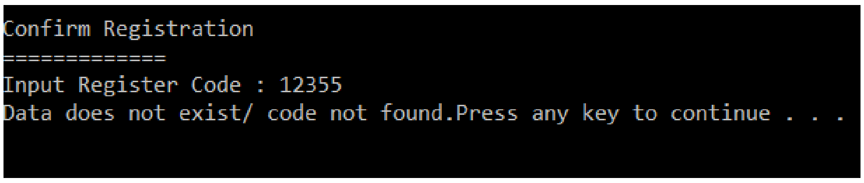


```
Confirm Return
=============
Input Return Code : GZ076
Item Code : TG070
Item Title : Horror 101
Item Type : Book
User Code : Database
User Name : user
User Address : Database
Date Borrowed : 23/6/2018

Confirm? [Y/N] : Y
Return success!
Book returned in time.
Press any key to continue . . .
```

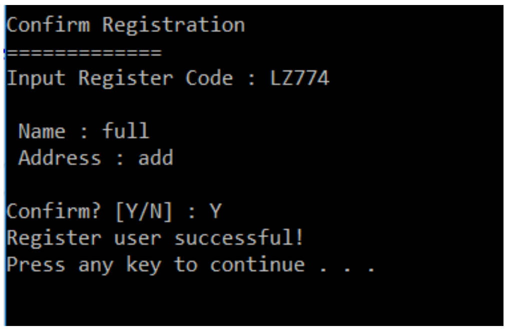
When found however, it will show all info about the item, and then ask for confirmation to return it. The program then will decide if the return is late or not (we built it to have a 2 week deadline), if it's late it'll show the fine for the charges (we made it $1 a day).

If the item is found on **Item Database**, it'll increase the stock by one, if not found, it'll make a new item to the **Item Database**. It'll delete the specific data on **Borrowed Database** and **Return Database**.

CONFIRM REGISTRATION



```
Confirm Registration
=============
Input Register Code : 12355
Data does not exist/ code not found.Press any key to continue . . .
```



```
Confirm Registration
=============
Input Register Code : LZ774

 Name : full
 Address : add

Confirm? [Y/N] : Y
Register user successful!
Press any key to continue . . .
```

On confirm registration option, it'll ask for the code that's displayed when user registered. when found, it'll show the registree information, and ask for confirmation. When confirmed, register the user to **User Database** (as a library user), and it'll delete the data on **Register Database**. The registree has fully been registered.

OOP APPLICATION

How the OOP applied in this program, is based on the databases. There are 5 databases, and they are :

- **ITEM DATABASE**
- **USER DATABASE**
- **REGISTER DATABASE**
- **BORROWED DATABSE**
- **RETURN DATABASE**

All of these, have specific class, which on read will define its objects. They have the user-defined setter getter function and things. on the **Item Database**, we have built a three-way class that defines each type of an item (book has author & ISBN; magazine has edition & ISSN; disc has owner & volume) that will inherit to a core class (essentially, like a fork). We also define overloading in the code which cannot be seen in the program.