LIMEWIRED

USER

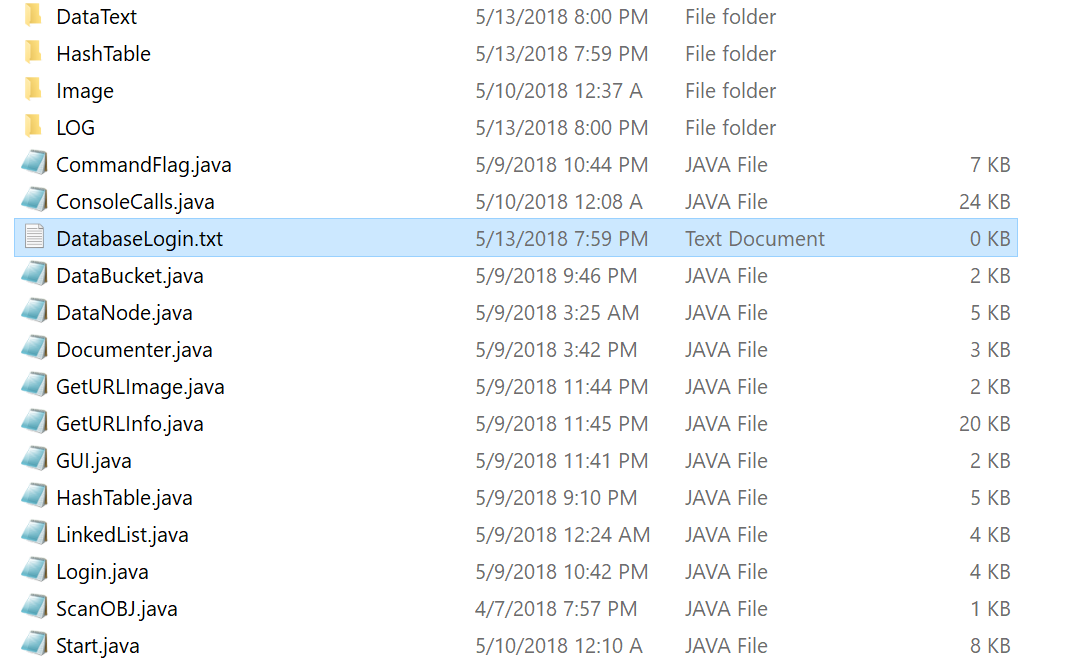
MANUAL

Steven Bedoya

**Table of Contents**

1. Installation
2. Login/Register
3. Adding/Searching
4. Finding
5. Deleting
6. Editing
7. Command Flags
8. EXTRA

**Installation**



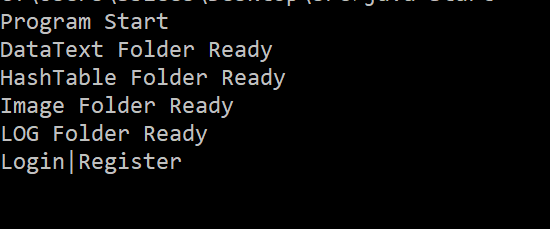
Step 1:

Make sure all provided java files and folders and DatabaseLogin.txt are in the same folder.

TIP: The folders DataText, HashTable, Image and LOG are checked at the start of the program to see if the exist if they don't then the program will create them.

Step 2: CD into the folder where all the contents are in and type in

* + javac Start.java
  + java Start

Once done you should see this.

And after your all set to start using LimeWired!

**To close the program when in the menu or login/register section type in EXIT(Not case sensitive).**

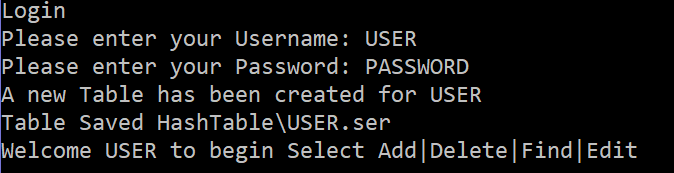
**To LOGOUT type LOGOUT when on the menu**

**if you don't see the menu type menu.**

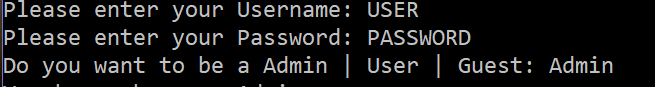
**Login/Register**

LOGIN

To login type login(Not case sensitive) and then input your Username and Password

TIP: If this is your first time signing in with a new username then a new table will be created for this user.

REGISTER

 To register type in register(Not case sensitive) and you should see this.

You will be first asked to type in a Username then Password and what level of permissions do you want to be.

TIP: What each level means

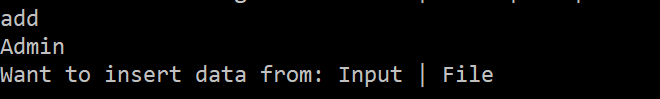
Admin/User: Can add/delete/edit data in the database

Guest: Can only search for data on the internet but can't add anything to the database.

**ADD**

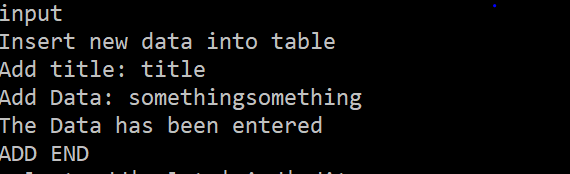
This section is used to search for data on wired/Pictures/Textfile and save it to the database/output to a textfile.

To start type in add(Not case sensitive)

This shows you your level of permissions and the options Input or File

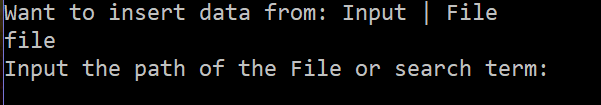
INPUT

INPUT is just a manual data input that allows you to add a title and a string of data. After entering your data, the data is saved into the hash table.



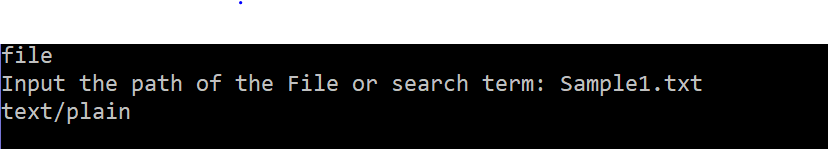
FILE

File is where you can add in textfile data, images off the web and search through data off of wired.com.

TIP: The program will automatically detect if the inputted data is a textfile, web picture link, wired link or if it is not either of those it will try an advanced search on wired.

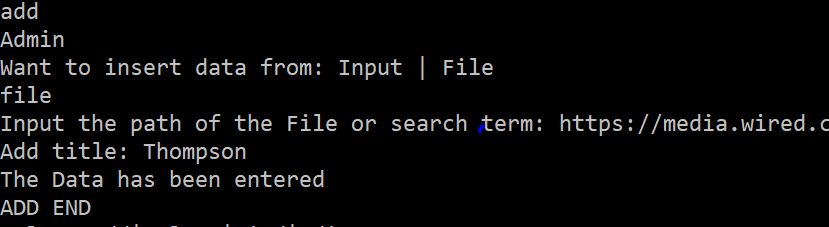
TextFile

To add text file information you will have to add the path of the text file.

PROBLEMS: After running the program after the final submission. I discovered a problem in which the program detects the text file and it never enters the data into the database and stays stuck as you can see in the picture so you will have to close the program. Sorry about this please don't try to add text file data.

Web Link Picture

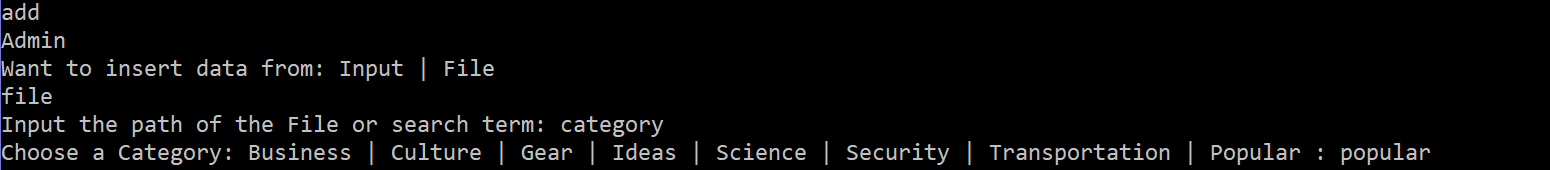
To add web picture just put in the web link after typing in file

After putting in a correct link it will make you add a title and then it will enter it into the database

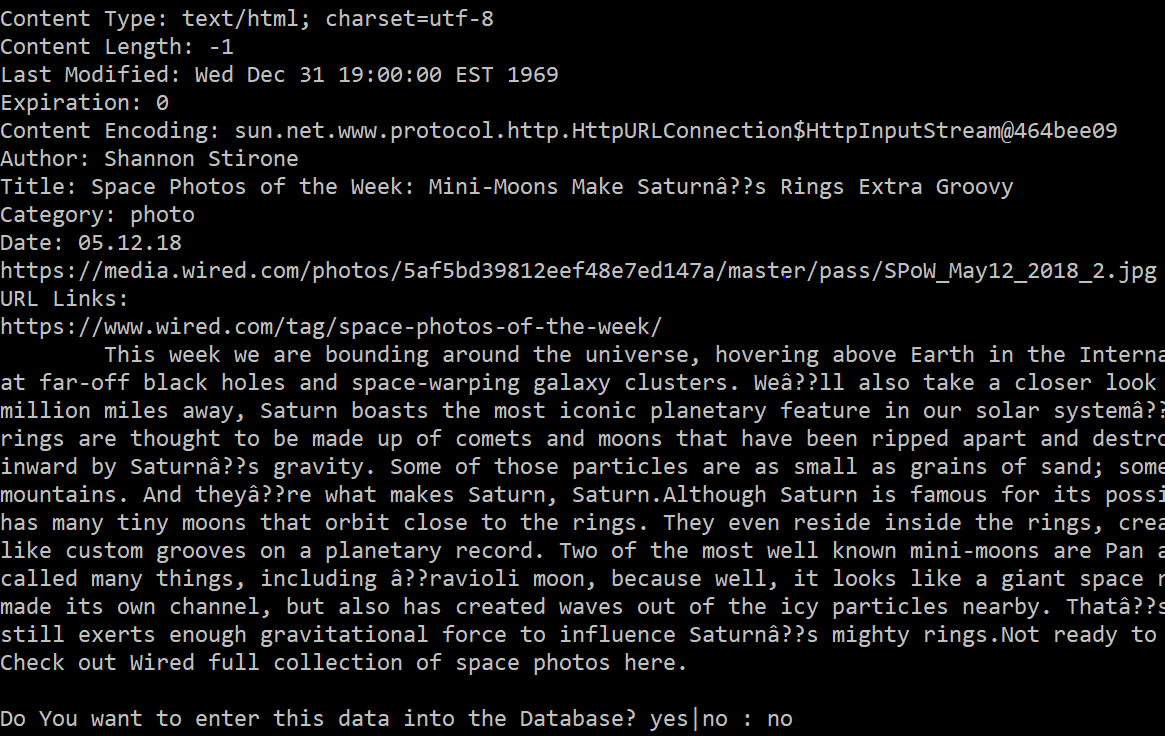
TIP: the picture is saved into the Image folder. Other non wired picture links can work too.

Category Search

To search by Wired category, type in add then file and then category.

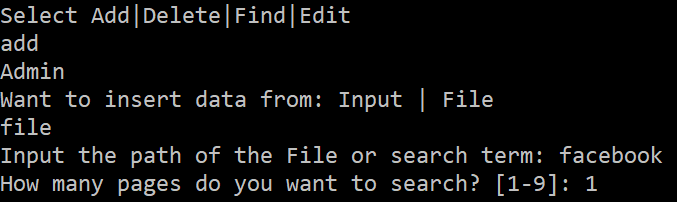
TIP: Once you choose one of the categories(Not case sensitive) then it will read off the first page of that choose category on wired.com. You will see all of the data being processed as it reads through the web page.

SAMPLE OUTPUT :

TIP: At the bottom it gives you the option to save the data into the database.

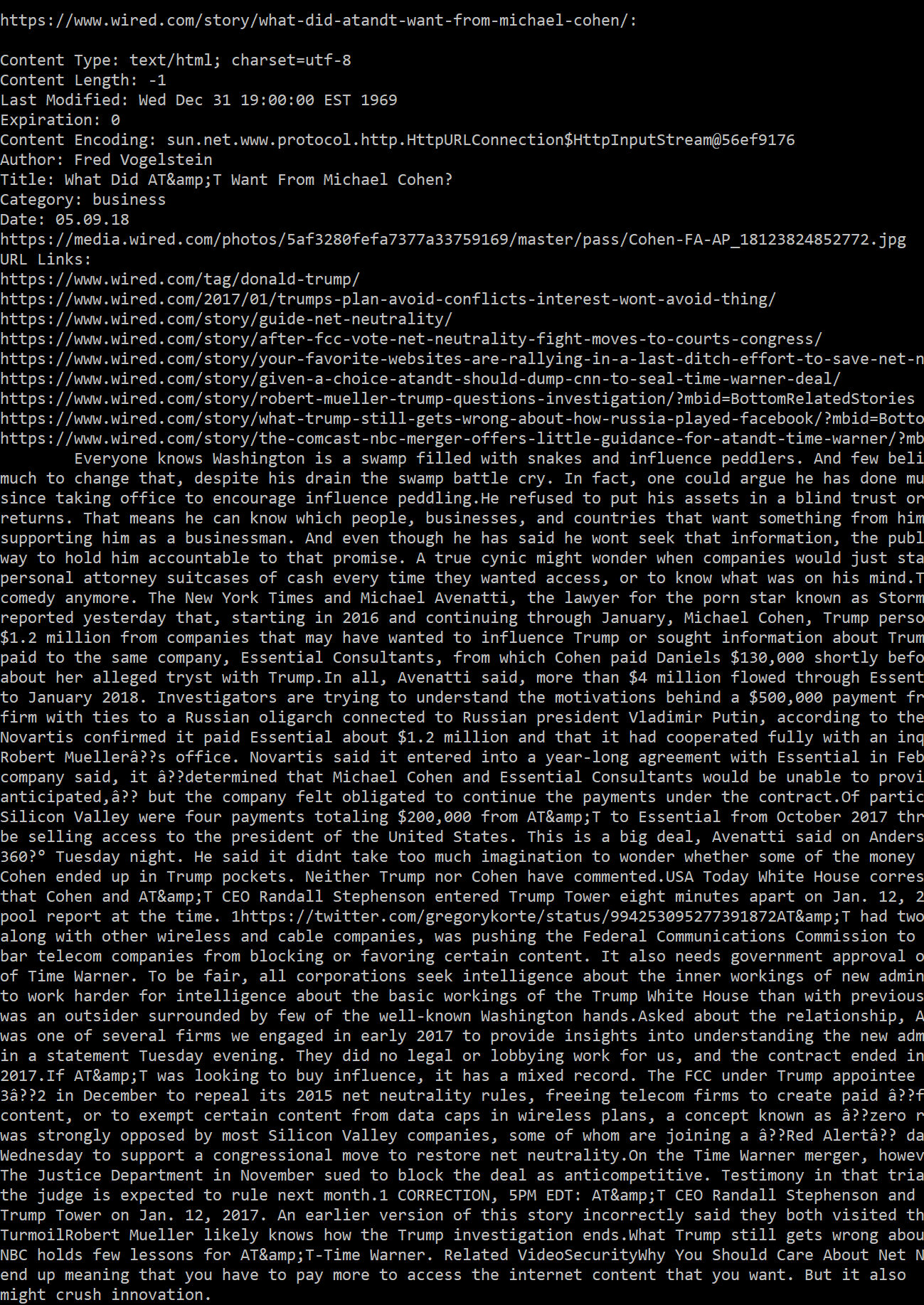
Advance Search

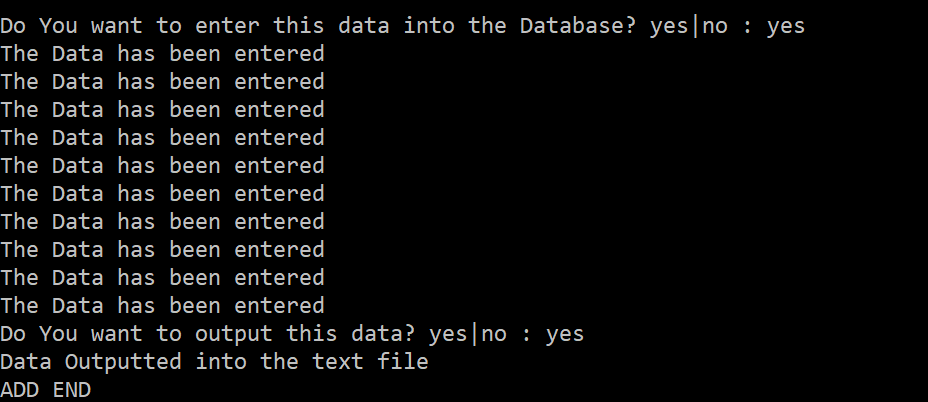
To do an advance search type in anything that is not a file path, web link or picture. Just type something like faceboook, cat, space, ai or phone.

TIP: The bottom part tells you how many search pages you want to to search. I recommend doing only doing 1-3 since it could take a while to search through a lot of data.

TECH ISSUE: Unlike inputting picture links which save the files into the Image folder doing advance search or a regular web link won't put it into the the folder but it does still saves it into the hash table and can be opened up later as you will see in the find section. I forgot to redirect the images into the image folder sorry.

SAMPLE OUTPUT NEXT PAGE.

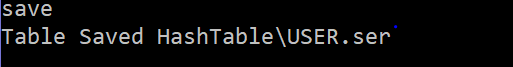


TIP: If you type yes you will see “the Data has been entered” which means the data has successfully entered the database.

The next section says “Do You want to output this data” is an option so that you can output the data found from the advance search into a output text file.

SAVE TABLE

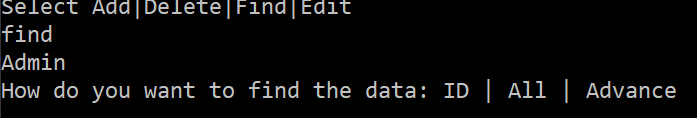
To save the table simply type in save(Not case sensitive).

TIP: Once you exit or logout the program it will tell you if you want to save. You need to save in order for the data to be available once you re open the progtam some other time.

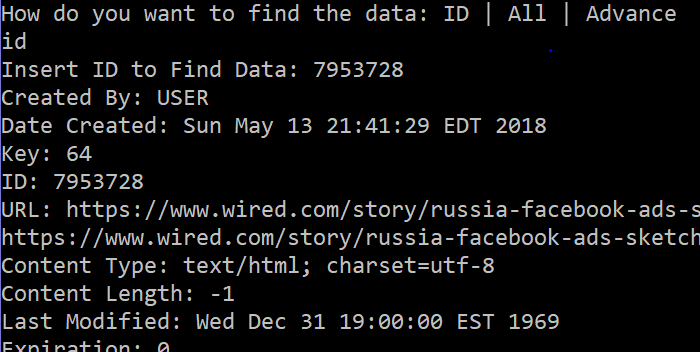
**REMEMBER TO TYPE SAVE TO SAVE YOUR WORK**

**FIND**

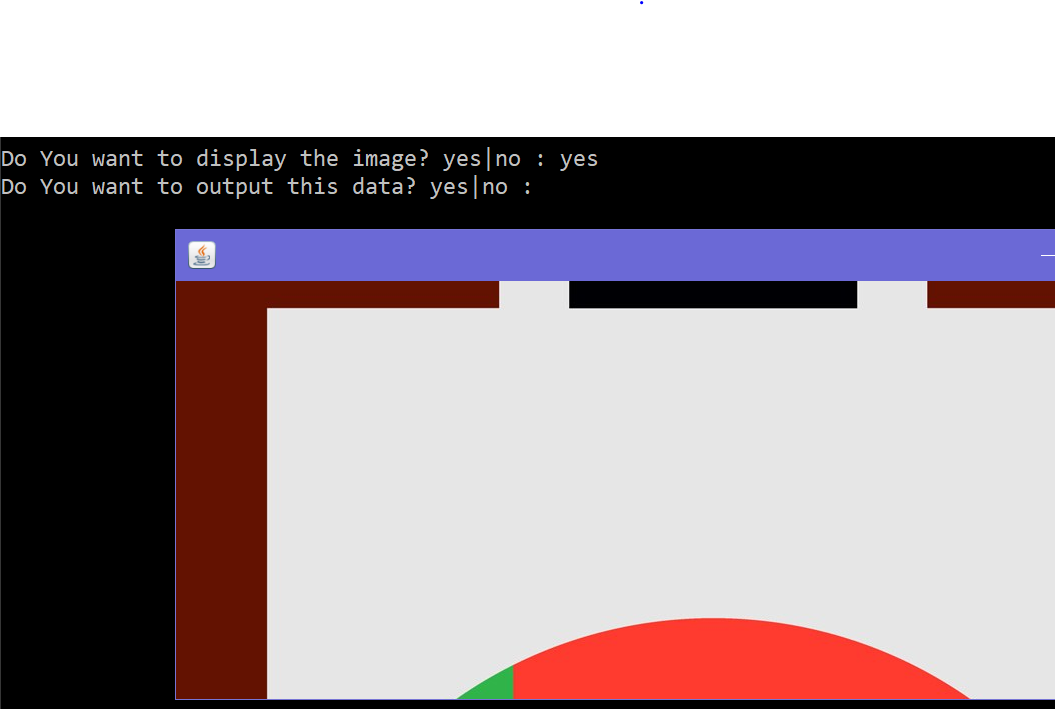
Find is used to find any data that is saved in the database.

ID

Every Data inputted as a special ID given to them and can be used to search for that data. This is a more accurate way in finding the data you want. Here is an example.

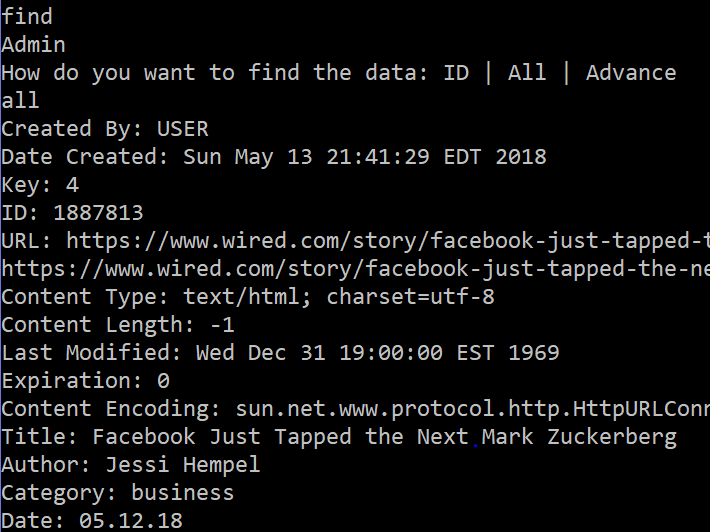
After finding it will give you the option to look at the picture and if you want to output the data to a text file. Next Page has example

TIP: Finding by ID is the only one with the option to look at the pictures since it works one at a time unlike the next find options which search through multiple data.



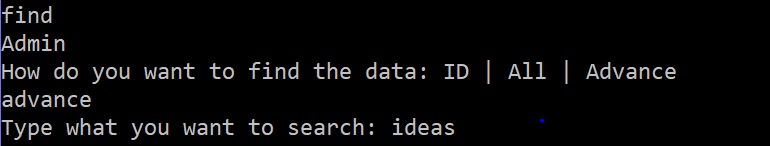
ALL

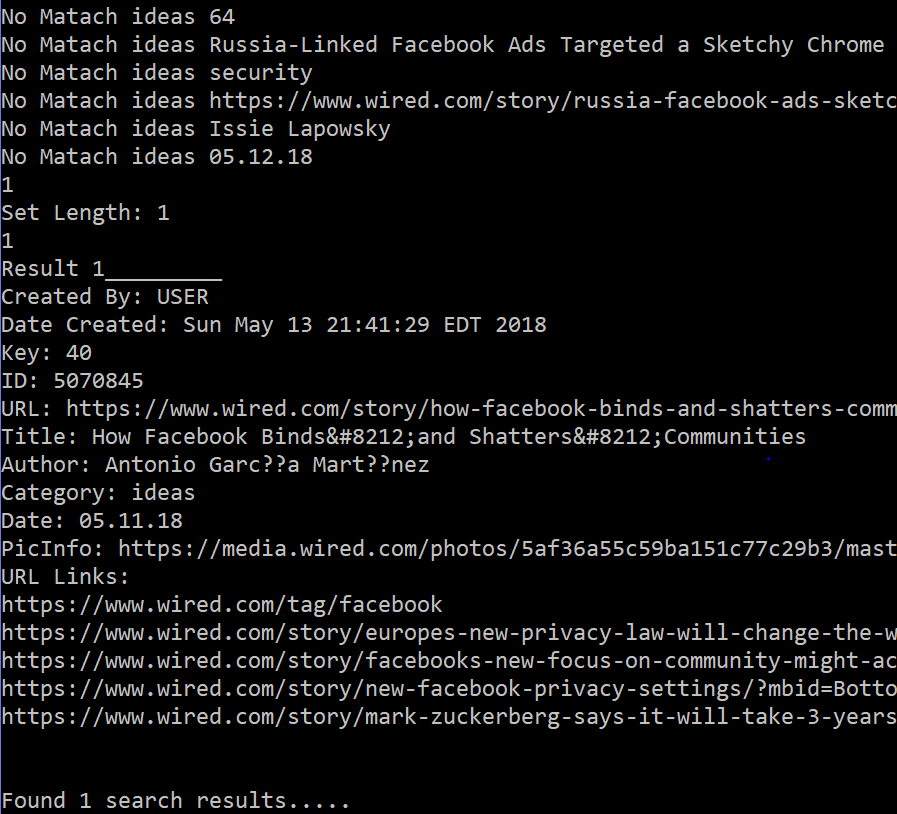
ALL outputs all the data in the database



Advance

Advance will search the data filtering out the data based on the inputted data.

SAMPLEOUTPUT:

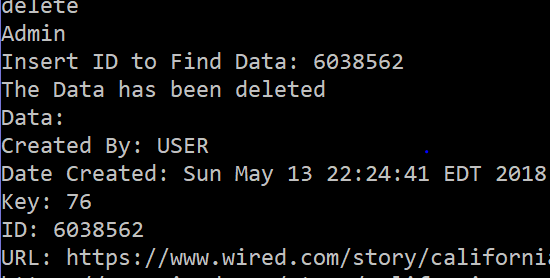
TECH ISSUE: The “No match.....” was used when testing and debugging the project and forgot to take it out.

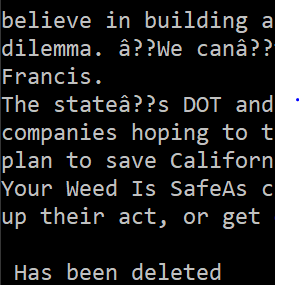
NOTE: The data such as the article text is not shown because I thought it would be easier to read the searched data in this format.

PROBLEM: If you add a web link picture into the database unlike a regular weblink or advance search then the program might throw an a null exception error. I found this out after the final submission and this is due to some of the data ending up being null when adding through a web link picture. It is an easy fix but didn't think of that issue before. This problem ONLY occurs in advance find.

**DELETE**

Delete removes data from the database and does it be searching for the data's ID number.



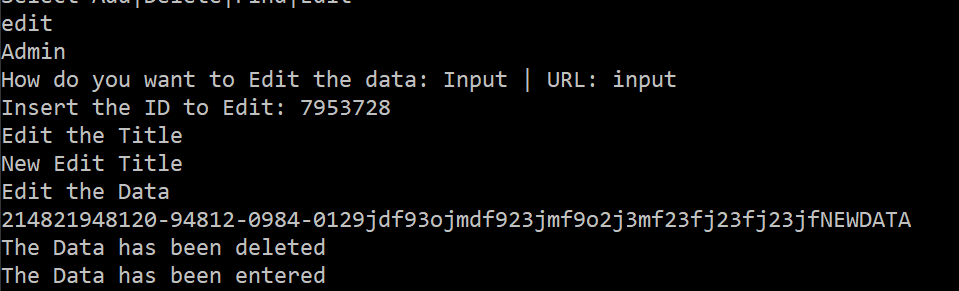
TIP: The easiest way to find the ID is by using the find ALL or find Advance and the copy and paste into the “Insert ID to Find Data:” section

**EDIT**

EDIT allows you to modify the data in the database and uses an ID number to find the data.

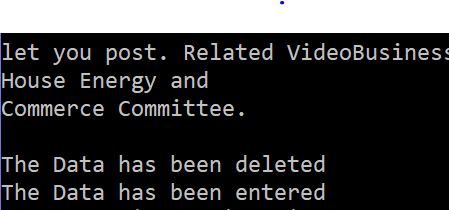
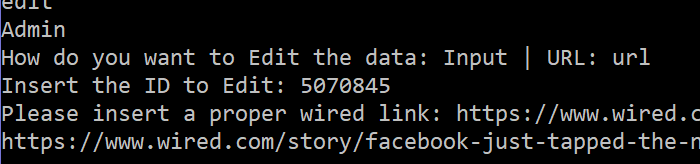
INPUT

INPUT allows you to manually change the datas title and text article data.

TECH ISSUE: “The data has been deleted” was going to be used for when a user wanted to edit the category a different hash number would be given to the data since the hashing works by category and date. I did not get to add the feature but the whole main purpose once to change the hash key but save the ID number.

URL

Replaces the data with new information from a wired link. Needs datas ID number in order to find and edit it.

TECH ISSUE: This where the “Data has been deleted” would have shine because the whole idea was to replace everything in the data node with new information but preserve the ID number. The thought process was if an article from wired was updated this could be used to update the new information. The ID is lost and is an easy fix but was too late after the final submission.

**Command Flags**

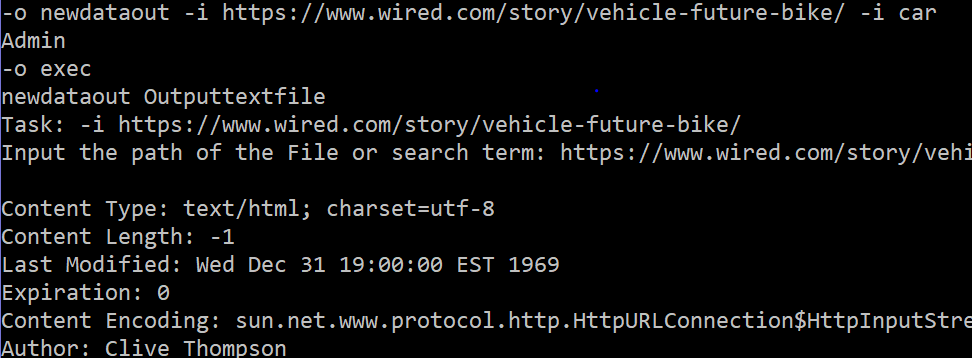
There are 4 command flags that could be used they are

-o: Used to output the data into a text file which comes from using -i.

EXAMPLE: -o [NAMEOFFILE]

TIP: Make sure to use this flag before using the -i flag since it will output all data that is called after -o is called.

EXAMPLE: -o {NAMEOFFILE] -i [WiredLink] -i [AdvanceSearchTerm]

-i: Works the same as the ADD->FILE Command which looks at what type of data is inputted and tries to find the data by link or advance search. Example shown above.

-f: Works use just like the find option which find the data in the database.

EXAMPLE: -f ALL Finds all data in the database

-f ID [IDNUMBER] Finds the data by ID Number.

-f advance [searchterm] Finds the data by the inputted data given.

-d: Deletes data from the database based on the ID numbers.

EXAMPLE: -d [IDNUMBER]

TIP: Multiple flags can be used in one line.

If the command prompt appears empty after doing the flags the program is still running to pull up the menu just type in menu.

**EXTRA**

* Saving data from the web can fill up pretty quick and even have large size from 100mb to 600mb! I still don't know much about how to optimize storage so the data doesn't take so much space.
* Having the login be saved in a text file runs a security risk and I wanted to do a different way but was focusing on making sure the online search worked. I made the login early in development and didn't learn of a better way until late in the project.
* When you open a picture up from using the find → ID. Clicking X on the picture closes the entire program so be warned. I didn't find out how to prevent that.
* The code that reads off the text file might have a bug or can't find the path correctly but it would work when I used it in my IDE.
* Sometimes the wired article text won't be read but it doesn't happen often and it is due to some of the articles not be consisted like the rest. But all of the data is still read.
* The database is not one united database. Every user has their own database structure(Hash table). Located in the HashTable folder.
* Hashing is done by hashing on the category then the month of the article.
* All logs are saved in the LOG Folder and are saved after closing the program or logging out.
* Outputted User Data is saved into the DataText folder with a folder of the user's user name.
* Ran out of time for the GUI so sorry.