USER MANUAL HDE

1- **Migrate Folder**
   - **a.** Run "bash bash1.sh" on command line by going to that directory. (This shell script code takes the HttpAnalyzer output and dumps into the database by updating some values.)
     - **i.** create_if_not_exist0.py checks the database and create HDE table if table doesn't exist.
     - **ii.** xls_to_csv1.py gets the all xls files in a path and converts those into csv files.
     - **iii.** csv_to_db2.py gets the all csv files and dumps the data into Mysql database.
     - **iv.** update_session_ids3.py updates the session id's when it sees 1 in sequence id.

2- **Scrape_data Folder**
   - **a.** Run "bash bash2.sh" on command line by going to that directory. (This shell script code gets the data from database and updates the indexed info of the URLs)
     - **i.** db_urlto_notepad4.py gets the URL colum from the database and write the column into a text file.
     - **ii.** Scrape.jar basically takes the every URL and searches on the bing search engine, gets the indexed info and writes into a csv file with "Url,Index(true,false)" format
     - **iii.** update_indexed5.py gets the output of scraper and updates the indexed column by looking only true values. We don't update every row, we update the value '0' if the index is true.

3- **Queries Folder**
   - **a.** Query1.py takes the session id from user and outputs the number of indexed and unindexed URLs
   - **b.** Query2.py
     - **i.** Takes the session id from user
     - **ii.** Finds the every consecutive URLs, that are in the url_crawler table, in hde table. That means these urls are crawled.
     - **iii.** Takes the consecutive id from the user

iv. Finds the corresponding id's of consecutive id's in the crawler table
v. Links to the link table and searches only one level of childs of the consecutive urls and outputs two set of lists
vi. Finds the common URLs in two lists and outputs the information of these URLs.

c. Query3.py
   i. Takes the session if from user
   ii. Finds the all urls are indexed and exists in the crawler table
   iii. Returns all the one level of childs of the indexed URLs in the hde table
   iv. Returns only the unindexed ones information

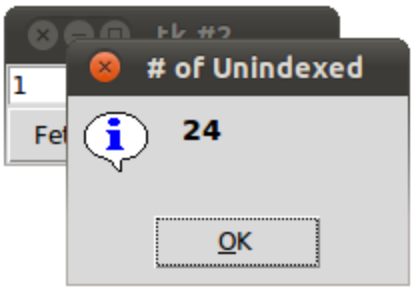
d. Query4.py
   i. Takes all sessions from database
   ii. Counts the number of indexed and unindexed URLs in each session
   iii. Takes to roots of sessions and checks whether if the root is a search engine or not

Queries screenshots

Query1

1- Pick the session id 1-2

2- Number of the Indexed and Unindexed of the selected session



Query2

1-Pick the session id from the list 1-2.

```
doga@doga-VirtualBox:~/Desktop/queries$ python query.py
1
2
PICK A SESSION ID FROM THE LIST:
```

2- Pick the pair number 1 (29,30), 2 (30,31), 3 (35,36)

```
------------------all consecutive pairs--------------------
29 30
30 31
35 36
-----------------------------------------------------------
You will be asked which consecutive you want to analyze.
You can only enter one of these consecutive pairs
Pick one number and enter
1
2
3
ENTER A PAIR NUMBER:
```

3- Outputs: corresponding id's in the url_crawler table and those ids' childs (1 level), common urls, and their information.

```
------------------------------------------------------------
--- 1- found the consecutive in the url crawler table
353
--- 2- found the first level of childs of the consecutive urls in the link table
[354L, 356L, 357L, 1836L]
--- 1- found the consecutive in the url crawler table
6
--- 2- found the first level of childs of the consecutive urls in the link table
[7L, 8L, 354L]
we have common!
354 = 354
the info about this url (url, secured, tld, indexed, http_access):
(('http://www.cs.indiana.edu/classes/b551/final_exam_study.html', 'http', 'edu', 0L, '---'),
)
doga@doga-VirtualBox:~/Desktop/queries$
```

Query3

1- Pick a session from the list

```
doga@doga-VirtualBox:~/Desktop/queries$ python query.py
1
2
PICK A SESSION ID FROM THE LIST:
```

2- It gives you the sets of first level child urls that are connected to the certain indexed link and gives you the only indexed ones information. In this case in the first list one of the urls is indexed.

```
doga@doga-VirtualBox:~/Desktop/queries$ python query3.py
1
2
PICK A SESSION ID FROM THE LIST: 2
these URLs are indexed and crawled
(353L, 'http://www.cs.indiana.edu/classes/b551')
(6L, 'http://www.scheme.com')
lists of childs in first level and their information (only unindexed ones)
----set-----
((354L,), (356L,), (357L,), (1836L,))
----info-----
('http://www.cs.indiana.edu/classes/b551/final_exam_study.html', 'http', 'edu', 0L, '---')
('http://www.cs.indiana.edu/classes/b551/final_exam_study.html', 'http', 'edu', 0L, '200')
('http://www.cs.rpi.edu/~hendler/presentations/engelmore.pdf', 'http', 'edu', 0L, '200')
----set-----
((7L,), (8L,), (354L,))
----info-----
('http://www.scheme.com/search.cgi', 'http', 'com', 0L, '---')
('http://www.cs.indiana.edu/classes/b551/final_exam_study.html', 'http', 'edu', 0L, '---')
('http://www.cs.indiana.edu/classes/b551/final_exam_study.html', 'http', 'edu', 0L, '200')
```

Query 4

1- Finds the number of indexed and unindexed URLs of all sessions, output the root information and show that root URL whether a search engine or not.

```
doga@doga-VirtualBox:~/Desktop/queries$ python query4.py
the number of indexed URLs in the %d th session # 1
((26L,),)
the number of unindexed URLs in the %d th session # 1
((23L,),)
info of the root url of %d th session 1
 www.google.com
root is a search engine
------------------------------------------------------------
the number of indexed URLs in the %d th session # 2
((23L,),)
the number of unindexed URLs in the %d th session # 2
((14L,),)
info of the root url of %d th session 2
 www.google.com
root is a search engine
------------------------------------------------------------
```