

The University of Texas at Dallas

Information Retrieval

CS 6322.001

Project Report

GROUP 5

The University of Texas at Dallas

CS 6322.001

Information Retrieval

Project

TITLE: Historical Artifacts Search Engine

Students

Nikita Vispute, nikita.vispute@utdallas.edu

Arpita Dutta, axd170025@utdallas.edu

Krishan Kumar, krishan.kumar@utdallas.edu

Pawan Vaidya, pav180001@utdallas.edu

Arihant Chhajed, arihant.chhajed@utdallas.edu

NIKITA VISPUTE
ARPITA DUTTA
KRISHAN KUMAR
PAWAN VAIDYA
ARIHANT CHHAJED

NETID:NXV170005
NETID:AXD170025
NETID:KXK180034
NETID:PAV180001
NETID:ARC180006

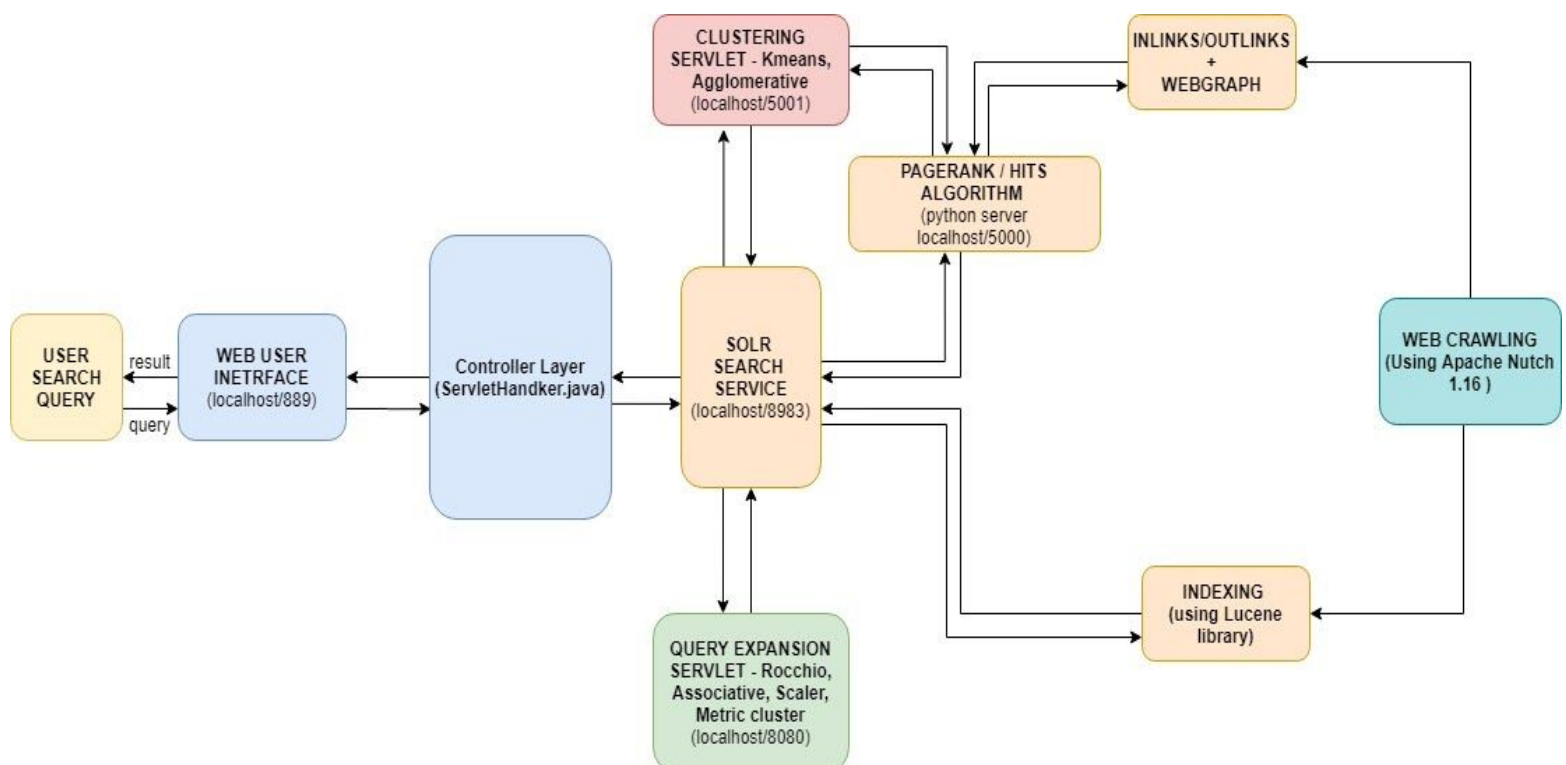
Introduction

This project report is based on the search engine we designed and built for Historical Artifacts.

The steps to build the search engine involved:

1. Crawling (Nikita Vispute)
2. Indexing (Arpita Dutta)
3. UI design (Krishan Kumar)
4. Clustering (Pawan Vaidya)
5. Query Expansion and Backend Design (Arihant Chhajed)

Architecture of the Search Engine:



CRAWLING

TECHNOLOGY STACK USED:

1. Apache Nutch-1.16
2. Solr-7.3.1
3. Selenium-2.42.2
4. Firefox
5. Java 8.0

APACHE NUTCH

- It is an Internet search engine software, web crawler, powerful for vertical search engine.
- The Apache Nutch Framework was utilized for
 1. crawling websites containing information on historical artifacts.
 2. for feeding the fetched crawled data to the Solr Framework for indexing. The Solr framework which is hosted on localhost:8983 port is used not only for indexing but also for creating web graphs that would later be used in implementing the Page Rank and the HITS algorithms.

Advantages of using Apache Nutch:

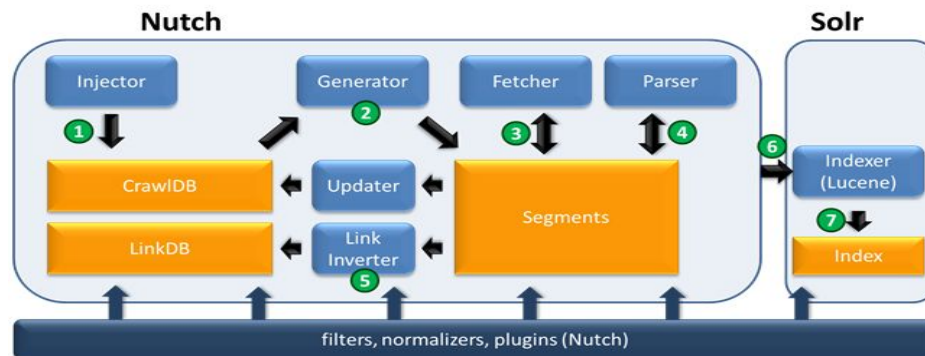
- Production ready Web Crawler, Scalable , Tried and Tested
- Fine grained Configurations
- Relying on Apache hadoop data structure, batch processing
- MultiThreaded.
- Allows Custom Implementation for parse, index and scoring.
- Pluggable Indexing (solr, mongodb, elastic search, etc)
- Automation on checking broken links
- Handle Duplication
- User friendly ,Url Filters, Normalization

APACHE NUTCH CONFIGURATION

Modifications were made to the nutch configuration file

Path: apache-nutch-1.1.16/conf/nutch-site.xml

1. Plugins: protocol-http|protocol-httpclient|urlfilter-regex|index-(basic|more)|query-(basic|site|url|lang)|indexer-solr|nutch-extensionpoints|protocol-httpclient|urlfilter-regex|parse-(text|html|msexcel|msword|mspowerpoint|pdf)|summary-basic|scoringopic|urlnormalizer-(pass|regex|basic)protocol-http|urlfilter-regex|parse(html|tika|metatags)|index-(basic|anchor|more|metadata)
2. Fetcher Server Delay (The number of seconds the fetcher will delay between successive requests to the same server): 4
3. Selenium Driver: firefox
4. Http Redirect Max The maximum number of redirects the fetcher will follow when trying to fetch a page. If set to negative or 0, fetcher won't immediately follow redirected URLs, instead it will record them for later fetching: 1
5. Ignore outlinks to the same hostname: False
6. Ignore outlinks to the same domain: False
7. Limit to only a single outlink to the same page: False
8. Selenium Hub Location connection port: 4444
9. Selenium Hub Location connection path: /wd/hub
10. Selenium Hub Location connection host: localhost
11. Selenium Hub Location connection protocol: http
12. A Boolean value representing the headless option for Firefox and Chrome drivers for Selenium : False
13. Boolean property determining whether the protocol-selenium WebDriver should capture a screenshot of the URL : False
14. The delay in seconds to use when loading a page with lib-selenium: 3
15. This number is the maximum number of threads that should be allowed to access a queue at one time: 2

NUTCH ARCHITECTURE:

The Crawling Process can be described by the repeating the following steps:

1. **Inject** (Seed list of size 70 unique urls)
2. Repeat for N Cycles :
 - a. **Generate** : Select URLs from Crawl DB for fetching.
 - b. **Fetch** : fetch URLs from fetch list.
 - c. **Parse** : extract content metadata and links
 - d. **UpdateDB**: Crawl Db status, score and signature, add new urls inlined or at the end of one crawler run.
 - e. **InvertLinks**: Map Anchor Text to document the link points to.
 - f. **Indexing** (Output of Invertlinks, given input to solr for indexing)
 - g. **DeDuplicate** Documents by Signature. (MD5 Hash is used)
 - h. **WebGraph** (creates webgraph of the crawled urls, inlinks, outlinks)

STEP 1: INJECTOR

1. The injector takes all the URLs of the seeds.txt file and adds them to the crawldb.
2. As a central part of Nutch, the crawldb folder maintains information on all known URLs (fetch schedule, fetch status, metadata).
3. After running this command, crawldb contains a list of unfetched URLs.

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch inject crawl/crawldb urls
```

STEP 2: GENERATOR

1. Based on the data of crawldb, the generator creates a fetch list and places it in a newly created segment directory.

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch generate crawl/crawldb crawl/segments -topN
```

STEP 3: FETCHER

1. The fetcher gets the content of the URLs on the fetch list and writes it back to the segment directory.
2. This step usually is the most time-consuming one as this respects and checks validity with the robots rules (robot.txt and robots directives)

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch fetch $s1
```

STEP 4: PARSER

1. Next the parser processes the content of each web page and for example omits all html tags.
2. If the crawl functions as an update or an extension to an already existing one (e.g. depth of 3), the updater would add the new data to the crawldb as a next step.

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch parse $s1
```

STEP 5: LINK INVERTER

1. Before indexing, all the links need to be inverted. This means the number of inbound links are considered, not the number of out-going links of a web-page.
2. This is quite similar to how Google PageRank works and is important for the scoring function.
3. The inverted links are saved in the linkdb folder.

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch invertlinks crawl/linkdb -dir crawl/segments
```

4. For creating the dump of segments containing the incoming and outgoing links for each URL.

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch readlinkdb Crawling/LinkDB -dump  
LinkDBDUMP
```

STEP 6 and 7: SOLR INDEXER

1. Using data from all possible sources (crawldb, linkdb and segments), the indexer creates an index and saves it within the Solr directory.
2. For indexing, the popular Lucene library is used.
3. Now, the user can search for information regarding the crawled web pages

Command:

```
Solr.bin/nutch index crawl/crawldb/ -linkdb crawl/linkdb/ crawl/segments/* -filter  
-normalize -deleteGone
```

STEP 8: DELETING DUPLICATES

- Duplicates (identical content but different URL) are optionally marked in the CrawlDb folder and are deleted later in the Solr index.
- Apache Nutch handles duplication of the crawled data via its properties setting and commands.
- By default Nutch uses the `org.apache.nutch.crawl.MD5Signature` class to calculate the digest of an URL, this class calculates the digest using the MD5Hash function of the raw binary content of the page, if no content is found then the URL is used.
- DeDup api of nutch is run in every crawl job, that handles duplicate urls fetched and removes them from crawl db. DeDup uses MD5 Signature to compare fingerprints of data fetched with existing data. If several entries share the same signature, the one with the highest score is kept. If the scores are the same, then the fetch time is used to determine which one to keep with the most recent one being kept. If their fetch times are the same we keep the one with the shortest URL. The entries which are not kept have their status changed to `STATUS_DB_DUPLICATE`, this is then used by the Cleaning and Indexing jobs to delete the corresponding documents in the backends. Hence, Dedup commands run after every dp update to handle duplicate data.

Command:

```
$NUTCH_RUNTIME_HOME/bin/nutch dedup <crawldb> [-group <none|host|domain>]  
[-compareOrder <score>,<fetchTime>,<urlLength>]
```

All these commands can be executed by running an automated bash script provided by nutch for big crawls located in `bin/crawls`. We used the command:

```
$NUTCH_RUNTIME_HOME/bin/crawl -i -s <path to urls/seed.txt> <output  
directory> <num of iterations>
```

We performed the crawling over a period of 4-5 days running continuously 10-12 hours over 4-5 iterations.

We managed to crawl around 204009 web urls and were able to generate 141909 documents for indexing after cleaning those crawled urls.

Apache Nutch generates 3 folders during the crawling operation:

1. CRAWLDB: it maintains the information about URLs such as the fetch status, fetching schedule, metadata, etc.
2. LINKDB: For each URL, this folder maintains the incoming and outgoing URLs to and from that URL. This information is later used to generate web-graphs and implement the PAGE RANKING algorithm and the HITS algorithm.
3. SEGMENTS: contains multiple subdirectories within it. During Crawling, the crawl script creates multiple directories to store information for Crawl Fetching, Crawl Content, Crawl Parsing, Parsed Data and Parsed Text.

STATISTICS FOR CRAWLED DATA:

The screenshot displays the Solr Admin UI for a Nutch instance. The left sidebar contains navigation links: Dashboard, Logging, Core Admin, Java Properties, Thread Dump, Overview (selected), Analysis, DataImport, Documents, Files, Ping, Plugins / Stats, and Query. The main content area is divided into three sections: Statistics, Instance, and Replication (Master). The Statistics section shows: Last Modified: about 4 hours ago, Num Docs: 141909, Max Doc: 154780, Heap Memory: -1, Usage: Deleted Docs: 12871, Version: 442, Segment Count: 30, and Current: ✓. The Instance section shows: CWD: /opt/solr-7.3.1/server, Instance: /opt/solr-7.3.1/server/solr/nutch, Data: /opt/solr-7.3.1/server/solr/nutch/data, Index: /opt/solr-7.3.1/server/solr/nutch/data/index, and Impl: org.apache.solr.core.NRTCachingDirectoryFactory. The Replication (Master) section shows a table with columns Version, Gen, and Size. The table has two rows: Master (Searching) with Version 1588101943616, Gen 58, and Size 2.41 GB; and Master (Replicable) with Version 1588101943616, Gen 58, and Size -. A Healthcheck section at the bottom right indicates: Ping request handler is not configured with a healthcheck file.

Version	Gen	Size
Master (Searching) 1588101943616	58	2.41 GB
Master (Replicable) 1588101943616	58	-

Description	Count
Total links crawled	204009
Number of links used in indexing after cleaning	141909

SEED LIST OF SOURCE WEBSITES:

Initially we gave **40 unique** domains as seed list URLs to start our crawling, later we added **30 more** unique domains.

We crawled websites from the domains like: joyofmuseums.com, getty.edu, metmuseum.org, britishmuseum.org, louvre.fr, wikipedia and similarly other domains. Most of these websites are those maintained by world famous museums around the world and specifically focus on the online collections of these museums. Some are also wikipedia websites.

The sample list of seed URLs are as follows:

1. <https://joyofmuseums.com/most-popular/most-popular-historical-objects/>
2. <https://joyofmuseums.com/most-popular/most-popular-sculpture/>
3. <http://www.getty.edu/art/collection/objects/333351/unknown-maker-funerary-relief-of-hadirat-katthina-daughter-of-sha'ad-palmyran-ad-200-220/>
4. <http://www.getty.edu/art/collection/objects/336764/unknown-maker-intaglio-with-bust-of-antinous-roman-ad-131-138/>
5. <http://www.getty.edu/art/collection/objects/336767/dioskourides-intaglio-with-bust-of-demosthenes-roman-about-25-bc/>
6. <https://joyofmuseums.com/portrait-guide/>
7. <https://joyofmuseums.com/most-popular/egyptian-art/>
8. <https://www.vam.ac.uk/collections?type=places>
9. <http://www.getty.edu/art/collection/objects/336737/unknown-maker-engraved-scaraboid-with-perseus-greek-400-350-bc/>
10. <http://www.getty.edu/art/collection/objects/327217/unknown-maker-applique-depicting-the-sun-god-usil-etruscan-500-475-bc/?dz=#3346028db30a5bd09b86640bd59a611f70fa30fb>
11. <https://joyofmuseums.com/most-popular/mythological-paintings/>
12. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/neues-museum/masterpieces-of-the-neues-museum/treasure-from-troy/>
13. <http://www.getty.edu/art/collection/objects/340870/giovanni-di-balduccio-the-annunciation-italian-about-1333-1334/>
14. <http://www.getty.edu/art/collection/objects/337384/veit-stoss-corpus-christi-german-about-1490-1500/>
15. <http://www.getty.edu/art/collection/objects/328670/auguste-rodin-bust-of-john-the-baptist-french-model-1880-cast-1886/?dz=#d92e3d69dcecc2e14fe3a7f36fcc5ce3aa0ba8c1>

16. <http://www.getty.edu/art/collection/objects/326677/desiderio-da-settignano-bust-of-a-young-boy-italian-about-1460-1464/?dz=#a1897d773b6c594712b391c5505ef6ea12c2cd82>
17. <http://www.getty.edu/art/collection/objects/294078/wouter-crabeth-the-prophet-habakkuk-and-the-angel-netherlandish-about-1565/>
18. <http://www.getty.edu/art/collection/objects/337412/juan-conchillos-falco-saint-john-the-baptist-spanish-november-9-1695/>
19. <http://www.getty.edu/art/collection/objects/328113/edme-bouchardon-hercules-subduing-the-centaurs-french-about-1735-1740/>
20. <https://joyofmuseums.com/museums/russian-federation/moscow-museums/pushkin-museum/>
21. <http://www.getty.edu/art/drawings/>
22. <http://www.getty.edu/art/collection/objects/266018/auguste-rodin-christ-and-mary-magdalene-french-1908/#15d6d66c35f59650befd7ef7ab0b86e76a741a1b>
23. <http://www.getty.edu/art/collection/objects/284651/gian-lorenzo-bernini-bust-of-pope-paul-v-italian-1621/?dz=#6dc75e5141ccf8566b8a3dc455fc424edd733c58>
24. <http://www.getty.edu/art/collection/objects/1138/fontana-workshop-possibly-orazio-fontana-or-possibly-flaminio-fontana-basin-with-deucalion-and-pyrrha-italian-about-1565-1575/>
25. <http://www.getty.edu/art/collection/objects/5806/beauvais-manufactory-woven-under-the-direction-of-philippe-behagle-after-cartoons-by-guy-louis-vernansal-et-al-tapestry-les-astronomes-from-l'histoire-de-l'empereur-de-la-chine-series-french-about-1697-1705/>
26. <http://www.getty.edu/art/collection/objects/226430/peter-paul-rubens-the-calydonian-boar-hunt-flemish-about-1611-1612/>
27. <http://www.getty.edu/art/collection/objects/333122/eustache-le-sueur-crucifixion-french-about-1650-1655/>
28. <http://www.getty.edu/art/antiquities/>
29. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/altes-museum/>
30. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/the-pergamon-museum/>
31. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/the-pergamon-museum/orpheus-mosaic-from-miletus/>
32. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/the-pergamon-museum/lion-hunt-relief-from-nimrud/>
33. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/the-pergamon-museum/highlights-of-the-pergamon-museum/the-pergamon-altar>

34. <https://joyofmuseums.com/museums/europe/germany-museums/frankfurt-museums/stadel-museum/>
35. <https://joyofmuseums.com/museums/europe/germany-museums/berlin-museums/deutsches-historisches-museum/fragment-of-the-heliand/>
36. https://www.metmuseum.org/art/collection/search/38882?searchField=All&sortBy=Relevance&showOnly=openAccess&ft=*&offset=60&rpp=20&pos=65
37. <https://www.metmuseum.org/art/collection/search?department=4&showOnly=highlights&offset=0&pageSize=0&sortBy=Relevance&sortOrder=asc&perPage=100>
38. <https://www.metmuseum.org/about-the-met/curatorial-departments/asian-art/highlights>
39. <https://www.metmuseum.org/about-the-met/curatorial-departments/european-sculpture-and-decorative-arts>
40. <https://joyofmuseums.com/museums/europe/greece-museums/athens-museums/the-acropolis-museum/>
41. <https://joyofmuseums.com/museums/europe/greece-museums/olympia-museums/archaeological-museum-of-olympia/>
42. <https://joyofmuseums.com/most-popular/popular-historical-sites/>
43. <https://joyofmuseums.com/most-popular/most-popular-sculpture/>
44. <https://www.metmuseum.org/art/collection/search/548362>
45. <https://www.louvre.fr/en/selections/jewelry>
46. <https://www.louvre.fr/en/selections/masterpieces>
47. <https://joyofmuseums.com/museums/europe/greece-museums/athens-museums/national-archaeological-museum-athens/>
48. <https://www.britishmuseum.org/collection>
49. <https://www.britishmuseum.org/collection/galleries/sudan-egypt-and-nubia>
50. <https://www.si.edu/explore/history>
51. <http://www.museivaticani.va/content/museivaticani/en/collezioni/capolavori.html>
52. <https://historyexplorer.si.edu/artifacts>
53. <https://www.npm.gov.tw/en/Article.aspx?sNo=02000019>
54. <https://theme.npm.edu.tw/selection/Category.aspx?sNo=03000130&lang=2>
55. <https://theme.npm.edu.tw/selection/Category.aspx?sNo=03000131&lang=2>
56. <https://theme.npm.edu.tw/selection/dynasty.aspx?idx=33&lang=2>
57. <https://theme.npm.edu.tw/selection/dynasty.aspx?idx=1&lang=2>
58. <https://www.khm.at/en/moments-objects-stories/>
59. <https://www.nationalgallery.org.uk/paintings/must-sees>
60. <https://www.vasamuseet.se/en/collections/search-the-collection-digitaltmuseum>
61. <https://www.vasamuseet.se/en/collections/search-the-collection-digitaltmuseum>
62. <https://www.theacropolismuseum.gr/en/content/gallery-slopes-acropolis>

63. <https://www.smb.museum/en/museums-institutions/pergamonmuseum/collection-s-research/collection-highlights.html>
64. <https://www.yadvashem.org/artifacts/museum.html>
65. <https://www.miaegypt.org/en-us/museum/collection/Collection>
66. <https://fotoinventari.uffizi.it/it/>
67. <https://www.namuseum.gr/en/collections/>
68. <http://www.ebyzantinemuseum.gr/?i=bxm.el.collections&c=2>
69. https://www.benaki.org/index.php?option=com_collections&view=collection&id=9&Itemid=162&lang=el
70. https://www.benaki.org/index.php?option=com_collections&view=collection&id=1&Itemid=162&lang=el
71. https://commons.wikimedia.org/wiki/National_Museum_of_Korea
72. <https://www.tobikan.jp/archives/collection.html>
73. https://en.wikipedia.org/wiki/Tokyo_National_Museum
74. <https://webarchives.tnm.jp/imgsearch/>
75. <https://www.museum.go.kr/site/eng/relic/recommend/list>
76. <http://www.nfm.go.kr/english/subIndex/971.do>
77. <https://www.nms.ac.uk/explore-our-collections/?subject=13102>
78. <https://www.nms.ac.uk/explore-our-collections/?subject=13126>
79. <https://www.nms.ac.uk/explore-our-collections/?subject=13130>
80. <https://www.ngv.vic.gov.au/explore/collection/curatorial/asian-art/>
81. <https://www.museum.go.kr/site/eng/relic/search/list>
82. https://en.wikipedia.org/wiki/National_Gallery_Singapore
83. https://en.wikipedia.org/wiki/Scottish_National_Gallery
84. https://en.wikipedia.org/wiki/Royal_Castle,_Warsaw
85. https://en.wikipedia.org/wiki/Gyeongju_National_Museum
86. https://en.wikipedia.org/wiki/National_Museum,_Krak%C3%B3w
87. <https://mnk.pl/collection>
88. https://www.heritagemuseum.gov.hk/en_US/web/hm/collections/introduction/chineseantiquities.html
89. <https://www.farandwide.com/s/best-museums-europe-70964e79d7b44d85>
90. <https://www.dpm.org.cn/explore/collections.html>
91. <https://www.vam.ac.uk/collections/south-asia>
92. <https://www.vam.ac.uk/collections/korea>
93. https://en.wikipedia.org/wiki/Ain_Sakhri_Lovers
94. https://en.wikipedia.org/wiki/Alligator_drum
95. https://en.wikipedia.org/wiki/Amman_Citadel_Inscription
96. https://en.wikipedia.org/wiki/Amulet_MS_5236
97. https://en.wikipedia.org/wiki/Anglo-Saxon_brooches

98. https://en.wikipedia.org/wiki/Antikythera_mechanism
99. <https://www.museodelprado.es/en/the-collection>
100. <https://www.nga.gov/collection.html>
101. <https://www.nationalgallery.org.uk/>
102. <https://www.toptenz.net/top-10-most-important-historical-finds.php>
103. <http://www.bbc.co.uk/ahistoryoftheworld/exploreraltflash/>
104. <https://www.ancient-origins.net/artifacts-other-artifacts/ten-amazing-artifacts-ancient-world-002105>
105. <https://www.livescience.com/57690-amazing-archaeological-discoveries.html>
106. <https://boredomtherapy.com/famous-historical-objects/>
107. <https://www.businessinsider.com/disputed-ancient-artifacts-2011-7>
108. <https://www.cnn.com/travel/article/best-usa-odd-world-history/index.html>
109. <https://www.factinate.com/things/44-amazing-facts-historical-artifacts/>
110. <http://www.state.in.us/dnr/historic/images/archsite.pdf>
111. <https://www.oldest.org/culture/artifacts/>
112. <https://www.nationalgeographic.org/encyclopedia/archaeology/>
113. <https://www.dhr.virginia.gov/archaeology/>
114. <https://www.wvcc.edu/library/assignment-topic-guides/historical-timelines-and-artifacts/>
115. <https://www.in.gov/dnr/historic/images/archsite.pdf>
116. <https://www.forbes.com/sites/kristinakillgrove/2015/06/12/five-reasons-you-shouldnt-buy-that-ancient-artifact/#32fd57f0a979>
117. http://usbizdata.com/lists/us-museums-and-art-gallery-list.php?gclid=CjwKCAjwv4_1BRAhEiwAtMDLsiohIVluTQAqQ76AZLyCRTLJXpQ2fGVosb7WjdZBPTDydWuBJnonTxoCR80QAvD_BwE
118. <https://www.wisconsinhistory.org/Records/Article/CS4050>
119. <http://www.ancientresource.com/>
120. <https://alj.artrepreneur.com/archaeological-treasures/>
121. <https://listverse.com/2014/11/08/10-authentic-historical-artifacts-no-one-can-explain/>
122. <http://www.bbc.co.uk/ahistoryoftheworld/>
123. <https://history.house.gov/Collection/Search?Classification=Historical%20Artifacts>
124. <https://www.ice.gov/factsheets/cultural-artifacts>
125. <https://www.nomadicmatt.com/travel-blogs/ten-historical-sites/>
126. <https://www.britannica.com/topic/World-Heritage-site>

HYPERLINK INFORMATION FOR INDEXING AND RELEVANCE MODEL

- Crawlddb folder of apache nutch has all the data that has been crawled.
- For each URL, the linkdb folder maintains information about the incoming and outgoing URLs to and from that URL.
- This information is extracted using the command
`$NUTCH_RUNTIME_HOME/bin/nutch readlinkdb <linkdb> -dump <out_dir>`
- After running the command, the output is a text file which is used as input for a python script written to extract relevant information like incoming and outgoing links for each URL as well as make a webgraph from the same for the indexing part of computing page-rank and HITS.

DISCUSSION ON CHALLENGES FACED DURING CRAWLING PROCESS:

1. Choosing correct version for Apache Nutch and Installation issues:
 - I started with Apache 1.15 installation on Windows OS. Initially the first two crawls carried out successfully, after which I had some corrupt issues with my Nutch installation.
 - After that I tried installing Apache 1.16. Multiple tries to install the software on Windows laptop failed due to hadoop and JAVA errors which could get resolved. My machine crashed while trying to configure Apache 1.16.
 - Since I was unable to install Nutch on my system, I started working on crawling the websites using Scrapy in python and this worked successfully.
 - After implementing the crawler with Scrapy and BeautifulSoup initially, the issue faced was extracting dynamic content (JavaScript/Ajax) from the websites automatically. For this I coded separate spiders for every url in the seed list.
 - But it was tedious to get web-graph and other details required for indexing through Scrapy as well. Also dynamic content web-scraping was running slow while using Scrapy.
 - As a last resort, pairing along with one of the team members I was able to install Nutch 1.16 and Solr framework on their machine on Ubuntu OS using Oracle VirtualBox and hence we shifted to the nutch framework and completed the crawling and indexing.
2. Nutch does not fetch AJAX/JavaScript driven dynamic HTML content.
 - In order to crawl web pages that rely on JavaScript/AJAX to dynamically load content we used the Selenium.
 - Since Selenium is a little slow we ran 5 iterations of the crawler continuously and ran it for 4-5 days.

3. Setup of Nutch 1.16 and Solr 7.3.1 on Amazon EC2 instance. Due to limited resources on free-tier instance, crawling failed after sometime because of out-of-memory issues.

```

FetcherThread 86 fetching https://mnk.pl/edukacja-w-mnk (queue crawl delay=2000ms)
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2498, fetchQueues.getQueueCount=90
FetcherThread 79 fetching https://mnk.pl/kontakt/oddzialy (queue crawl delay=2000ms)
FetcherThread 42 fetching https://www.heritagemuseum.gov.hk/en_US/web/hm/collections/introduction/contemporary.html#top (queue crawl delay=2000ms)
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2500, fetchQueues.getQueueCount=93
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2500, fetchQueues.getQueueCount=93
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2500, fetchQueues.getQueueCount=93
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2500, fetchQueues.getQueueCount=93
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2500, fetchQueues.getQueueCount=93
-activeThreads=50, spinWaiting=0, fetchQueues.totalSize=2500, fetchQueues.getQueueCount=93
Error running:
/home/ubuntu/apache-nutch-1.16/bin/nutch fetch -D mapreduce.job.reduces=2 -D mapred.child.java.opts=-Xmx1000m -D mapreduce.reduce.speculative=false -D mapreduce.map.speculative=false -D mapreduce.map.output.compress=true -D fetcher.timelimit.mins=180 output2//segments/20200426070237 -threads 50
Failed with exit value 137.

ubuntu@ip-172-31-38-0:~/apache-nutch-1.16$ ls -l output2
total 12
drwxrwxr-x 4 ubuntu ubuntu 4096 Apr 26 07:02 crawldb
drwxrwxr-x 3 ubuntu ubuntu 4096 Apr 26 07:01 linkdb
drwxrwxr-x 5 ubuntu ubuntu 4096 Apr 26 07:02 segments
ubuntu@ip-172-31-38-0:~/apache-nutch-1.16$ ls -l output2/linkdb/
total 4
drwxrwxr-x 3 ubuntu ubuntu 4096 Apr 26 07:01 current
ubuntu@ip-172-31-38-0:~/apache-nutch-1.16$ ls -l output2/linkdb/current/
total 4
drwxrwxr-x 2 ubuntu ubuntu 4096 Apr 26 07:01 part-r-00000
ubuntu@ip-172-31-38-0:~/apache-nutch-1.16$ ls -l output2/linkdb/current/part-r-00000/
total 1344
-rw-r--r-- 1 ubuntu ubuntu 1369927 Apr 26 07:01 data
-rw-r--r-- 1 ubuntu ubuntu 2014 Apr 26 07:01 index

```

Activate Windows
Go to Settings to activate Windows.

5. Filters

- Pages like blogs, dummy pages, help pages etc were being crawled which are irrelevant to the required data and hence were not crawled in the future by adding a regex url filter.

6. Selenium with Firefox compatibility issue:

- Selenium has an automated browser. Nutch loads its pages in selenium's browser. So, It's very important to find a compatible version of the browser with selenium. Versions mentioned in requirements are compatible versions with nutch 1.16.
- Selenium has a threading issue of memory leak: So due to this, many times the browser gets stuck on a page and does not terminate, thus throwing a port locked exception. Hence a command is run every 15-30 mins to kill firefox. This releases the locked ports, to be used by active threads.

SOLR:

- Every version of Nutch is built against a specific Solr version , So the compatible version with Nutch-1.16 is Solr-7.3.1 . We used Solr to monitor the data.
- To start Solr: `./bin/solr start`
- Admin Console: <http://localhost:8983/solr/#/>

SELENIUM:

- Nutch do not fetch AJAX/JavaScript driven dynamic HTML content
- In order to crawl web pages that rely on JavaScript/AJAX to dynamically load content we used the Protocol-Selenium Plugin.
- This plugin will load the pages that you're crawling in Selenium so that JavaScript will be handled properly.
- So we used selenium -2.42.2 with Nutch and Firefox-29 which is the compatible version.

COLLABORATION WITH INDEXING PERSON (Arpita Dutta)

- The indexing step is done as the last step in crawling hence the crawling and indexing was run together on the Nutch and Solr framework.
- The hyperlink information was extracted from the linkdb folder using the read linkdb dump command which gave the output as a text file and this was forwarded to Arpita who then wrote a python script to generate a web-graph for relevance models and link analysis.

CONCLUSION

1. Apache Nutch is highly scalable, robust and relatively feature rich crawler.
2. If we use Apache Nutch on AWS, Google Cloud Computing, it can be clustered on multiple machines at a time which can result in fast performance or it can be run in distributed mode using big-data mapreduce format.
3. In coordination with the Solr framework both crawling and indexing jobs can be done together thus easing the process.
4. Using the Selenium plugin and firefox driver we can also fetch and extract the dynamic content from the websites.

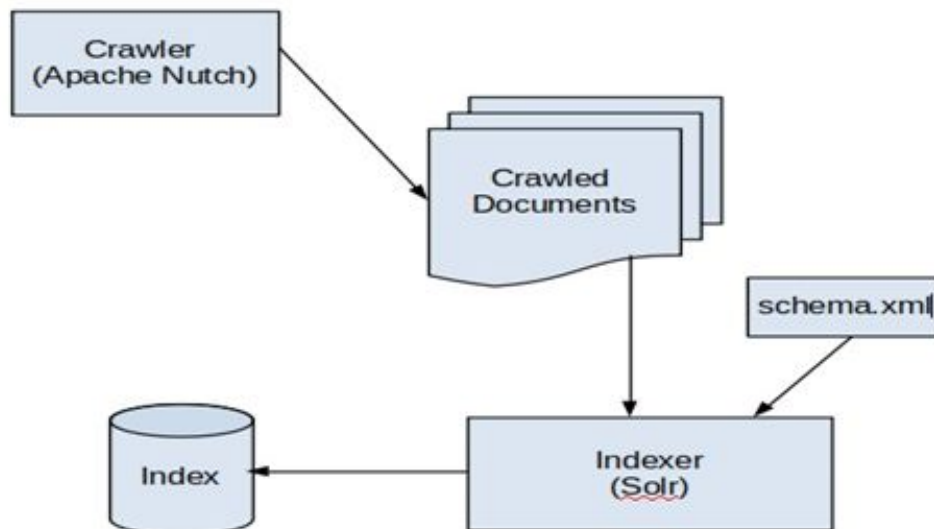
INDEXING AND RELEVANCE MODEL

TECHNOLOGY USED:

1. Solr 7.3.1
2. Apache Lucene Library

INDEXING

3. Indexing and relevance based retrieval is the most important aspect of any search engine.
4. It directly impacts the performance of the search engine.
5. Poor indexing and/or relevance models will result in slower and irrelevant results.
6. For our project, we used Solr for indexing. Solr is a high performance search server built using Apache Lucene core which provides indexing and other core information retrieval functionality.
7. For our search engine, following figure depicts overall indexing architecture:



8. Apache Nutch is used to crawl the websites in the seed list. After crawling is done, nutch stores the data in three folders named crawldb, linkdb and segments, as described in the crawling part. This data is then used by Solr Indexer to create the index.

9. Indexer uses schema.xml file. This file defines what kind of fields are present in the documents, which field should be used as a primary key to uniquely identify all the documents, and which fields are required to help in indexing and searching.
10. Before being added to the index, documents go through the analysis phase of the Indexer which pre-processes the documents by performing tokenizing, lower-casing, removing stop words and stemming/lemmatization etc. After that, we have tokens of each document which can then be added to the index.
11. The crawling script includes the automatic indexing step using the following command:
<http://localhost:8983/solr/#/>
bin/nutch solrindex http://localhost:8983/solr/nutch crawl/crawldb/ -linkdb crawl/linkdb/ -dir crawl/segments/ -deleteGone -filter -normalize
12. This is an example of incremental indexing. This means, as the documents are crawled simultaneously they are being indexed. This is one of the major reasons to use Nutch and Solr together as it saves a lot of time and is efficient.
13. Solr also allows easy merging of two separate crawls(both data and indexes) done on separate machines, this was helpful to us since two people were running the softwares.

LINK ANALYSIS

Webgraph

1. Apache Nutch Crawler does not generate a webgraph but it does generate a segments folder which stores actual content of the crawled and fetched documents as well as a linkdb folder that stores all the information on inlinks and outlinks for each URL.
2. This linkdb folder was extracted to a text file by Nikita Vispute using the readlinkdb dump command and sent to me.
3. I wrote a python script to parse this inlink and outlink data from the text file and make a web-graph from the same.
4. Some of the statistics for the webgraph are as follows:
 - Number of nodes: 117330
 - Number of links: 264243
 - Maximum number of in-links: 8427
 - Maximum number of out-links: 98

PageRank Algorithm:

1. Pagerank is a query independent link analysis algorithm that is executed at indexing time and it gives each document/URL a relevancy score based on a weighting scheme.
2. The Solr server uses the built-in page rank algorithm and displays the search results on the server. LinkRank is the implementation provided by Nutch and it uses the tf-idf weighting scheme.
3. In our project, after indexing and creating a webgraph, we further process it using the networkx library which dynamically creates a new directed graph from the webgraph for every search query.
4. After this the pagerank algorithm is applied using the built-in function in the networkx library: `networkx.pagerank()` and the score result returned is a dictionary of pagerank value and the url.
5. Parameters used: alpha value = 0.9
6. Top pages with highest page ranking are as follows:

https://en.wikipedia.org/wiki/Egyptian_pyramids	351.9641
https://en.wikipedia.org/wiki/Pyramid	345.50262
https://en.wikipedia.org/wiki/Great_Pyramids	265.01923

HITS Algorithm:

1. Hyperlink-Induced Topic Search (HITS) is a query dependent link analysis algorithm and thus is executed at search time.
2. It first dynamically creates a subgraph from the original webgraph based on pagerank results of the search query and assigns hub and authority scores to the documents in this subgraph.
3. The score results returned is a tuple of the hub and authority score and the ranking of these documents is based on the hub scores using the `networkx.hits()` built-in function. The sorted results are rendered onto the UI.
4. Parameters used: maxiterations:100 and tol:0.1

RELEVANCE MODELS

1. Apache Lucene built-in with Solr uses a scoring function which is a combination of the **Vector Space Model** (VSM) and **Boolean model** to determine how relevant a given Document is to a User's query.
2. In general, the idea behind the VSM is that the more times a query term appears in a document relative to the number of times the term appears in all the documents in the collection, the more relevant that document is to the query.

3. It uses the Boolean model to first narrow down the documents that need to be scored based on the use of boolean logic in the Query specification. Lucene also adds some capabilities and refinements onto this model to support boolean and fuzzy searching, but it essentially remains a VSM based system.
4. Thus we have used Apache Lucene (VSM and Boolean Relevance Models) built into Solr (Indexer) in combination with the Pagerank and HITS algorithms to show our query results.
5. Apache Lucene uses inbuilt TF-IDF based weighting scheme with cosine similarity between documents and queries.

Vector Space Model:

$$\text{Score}(D, q) = \frac{V(D) \cdot V(q)}{|V(D)| |V(q)|}$$

where, $V(D) \cdot V(q)$ is a dot product of weighted document and query vectors while $|V(D)|$ and $|V(q)|$ are their euclidean norms.

Weighted vectors are calculated with the tf-idf weighting scheme.

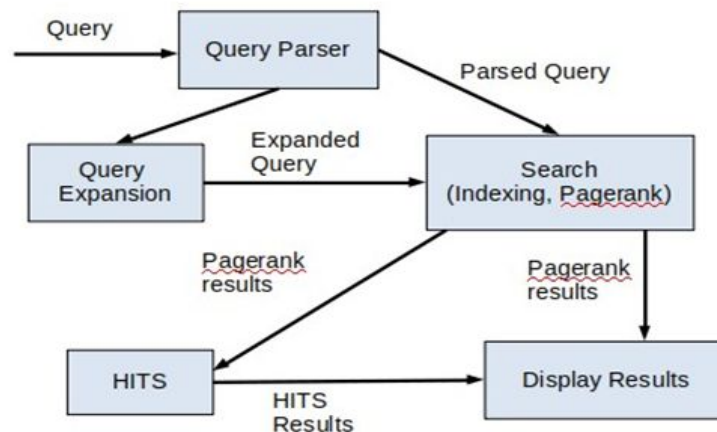
On top of these calculations, Solr also incorporates query boosts and document length normalization. More information on these boosts and Solr's (Lucene's) score functions can be found here:

https://lucene.apache.org/core/4_0_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html

Boolean Model:

- Boolean model is the most common exact-match model where queries are logic expressions with document features as operands.
- In the pure Boolean model, retrieved documents are not ranked. It is used to first decide which documents need to be ranked and which do not, prior to using the vector space model.

Our Retrieval architecture is depicted in a figure below:



Topic-based page ranking

I chose the following query for topic-based page ranking. They are broad enough queries that they can serve as a topic in the context of historical artifacts.

Topic: museum

Page rank

URL

COLLABORATION WITH UI PERSON (Krishan Duhan)

1. Data, indexes and pagerank scores are available in Solr server. So, information about Solr server and query syntax/GET request is passed on to Krishan Duhan.
2. Once the query results were available from the pagerank algorithm that are run on a python server, they were passed on to the UI as a json object. These results are already sorted according to pagerank scores so the UI reads the json object and displays the results.
3. Similarly, the python code for HITS algorithm runs a server to implement HITS algorithm. UI code gives the page-rank output as input to this server and gets the results back from HITS algorithm as a json object. These results are already sorted based on hub and authority score so UI just reads the json object and displays the results.

4. I used about 15 queries (mostly about) to test the results. Results were judged by manually checking contents of top 10 web pages returned by our search engine and comparing them to the search results returned for the same query by Google and Bing search engines.

COLLABORATION WITH CLUSTERING PERSON (Pavan Vaidya)

1. Pagerank results were passed on to Pavan Vaidya. Clustering module will look at these results and cluster the documents according to the clustering algorithms implemented and display the top documents from that cluster as results on the UI.
2. The clustering results are returned as a json object with top 12 documents of the optimum cluster value displayed.

DISCUSSION:

While implementing Scrapy with python we faced the issue of how the output from crawling should be generated and what can be used for indexing, but once we started using Nutch and Solr open source, this got resolved. Once the crawling started we discussed the type of output required from crawling so that indexing was read and processed on it.

CONCLUSION:

Once the crawler worked using Apache Nutch Solr was used for indexing which was easily integratable with Nutch. The main part where time was consumed was writing a script to extract the inlink and outlink information and create a web graph from the same so as to implement the pagerank and HITS algorithms.

USER INTERFACE AND GOOGLE/BING COMPARISON

TECHNOLOGY STACK USED

1. HTML5
2. JavaScript
3. jQuery
4. CSS
5. Bootstrap
6. Java Servlets
7. Tomcat Server

Design

The design part has four layers which are described below:

- Presentation Layer
- Application Layer
- Data Access Layer
- Business Layer

Presentation Layer - User interface

1. Front end is rendered using HTML5/CSS and Javascript library and is powered by using Java Servlets.
2. The backend is a dynamic web application in JAVA and consists of 4 JAVA servlets.
3. The JAVA controller makes the calls to all the various services in the back-end.
4. Bootstrap CSS library is used to design the page layout.
5. The search feature allows to query for various historical artifacts and corresponding statistics.
6. The results of the query are shown in different tabs.
7. The web search engine has the following components:
 - The Logo
 - A search bar
 - Navigable tabs to compare/contrast between different search results.

Following tabs are visible on the UI to choose from:

1. Google - Shows results from google, using google search API.
2. Bing - Shows results from Bing, using Bing Search API.
3. PageRank Algorithm.
4. HITS Algorithm.

5. Flat (K-means) Clustering.
6. Agglomerative Clustering (single/complete/average linkage).
7. Query Expansion using Rocchio algorithm.
8. Query Expansion using association clustering.
9. Query Expansion using scalar clustering.
10. Query Expansion using metric clustering.

Google Search Results using Google Search API:

1. Sourced from:
https://www.googleapis.com/customsearch/v1?key=INSERT_YOUR_API_KEY&cx=017576662512468239146:omuauf_lfve&q=lectures
2. Following are the 3 query parameters:
 - **API key** - Use the key query parameter to [identify your application](#).
 - **Custom search engine ID** - Use **cx** to specify the custom search engine you want to use to perform this search. The search engine must be created with the [Control Panel](#)
 - **Search query** - Use the **q** query parameter to specify your search expression. Placing a GET request to the above API returns results that are in json format.

Bing Search Results using Bing Search API:

1. Displays results from Microsoft Bing Search API.
2. Subscription Key
3. Custom Configuration Key
4. Custom Config ID

Pagerank Results:

Results displayed directly onto the UI from the python server as a json object as implemented on the documents retrieved from the Solr service at localhost/8983 after indexing is done and the relevance model using Apache Lucene has been implemented on it based on the search query.

HITS Results:

Results from the python server are displayed onto the UI as a json object after applying HITS algorithm on the pagerank results on documents retrieved from the Solr service based on the search query.

Flat Clustering:

Results from Clustering server at localhost/5001 after applying K-means or agglomerative clustering over Pagerank results according to user query are displayed onto the UI as a json object.

Query Expansion Results:

Results from the Query Expansion server at localhost:8080 after expanding the query using relevance feedback algorithms are displayed onto the UI as a json object.

Tomcat server is being used to handle the GET and POST requests used to send requests to the various servers at localhost:8089. There are 3 controller programs that are programmed as JAVA servlets.

Data Access Layer:

The request that we get from the front-end UI at [localhost/8989](http://localhost:8989) is routed to the corresponding server and then the result is given back to the Controller layer.

Business layer:

There are 4 servers running locally that contain the business logic for various relevance models.

1. **Server 1:** Solr service for handling indexing and search requests.
<http://localhost:8983/solr/#/>
2. **Server 2:** Python server for handling PageRank/HITS requests.
<http://localhost:5000/getHITS/<true/false>?+searchText>
3. **Server 3:** JAVA server for handling Flat and Agglomerative Clustering requests.
<http://localhost:5001/getCluster/clustertype>
4. **Server 4:** JAVA server for handling query expansion requests.
[http://localhost:8080/QueryExpansion?query="+query+"&qe="+type](http://localhost:8080/QueryExpansion?query=)

All requests from the Data Access layer are routed to these three servers. JSON is chosen as the data exchange format for ease of implementation and support by various programming languages.

COMPARISON WITH GOOGLE AND BING SEARCH ENGINES:

In order to compare the results with Google and Bing results, Google web search API and Bing Search API were used to retrieve their results. Following observations were made during the evaluation:

1. For certain queries which contained historical artifacts, our results from Pagerank/HITS/Clustering were closely in accordance with results from Google and Bing. We believe this is due to the selection of very good seed URLs to cover most of the food recipe possible.
2. Using seed URLs from Google and Bing proved useful in retrieving the top pages during the search. Wikipedia pages were immensely useful in improving our search results and were retrieved by our search engine despite the limited number of inlinks and outlinks in comparison to the entire web used by Google and Bing. This proved the effect of the Vector Space model combined with Pagerank.

TESTING STRATEGY:

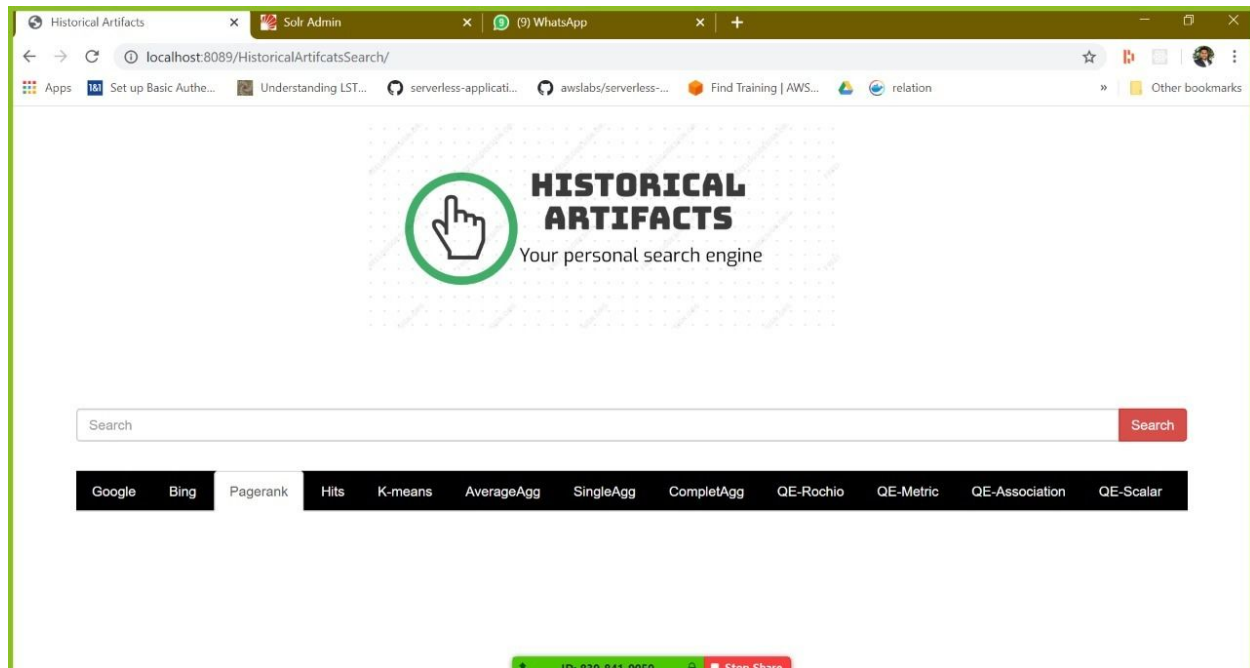
Testing was done throughout the development of the project to provide immediate feedback to students implementing the relevance models. Approximately 50 queries were used to test the results of various models. About 25 queries were used in collaboration with the students building the relevance models and the rest of queries were generated based on various topics. When a particular page from the top results of Google/bing does not come up with our relevance model, the page was first checked for existence and if present, the page rank score of the page was compared with the top scoring page from our relevance model. In a few cases, the score was too low due to insufficient inlinks and outlinks and in many cases, the page in question was not present in our collection.

KRISHAN KUMAR

NETID: KXK180034

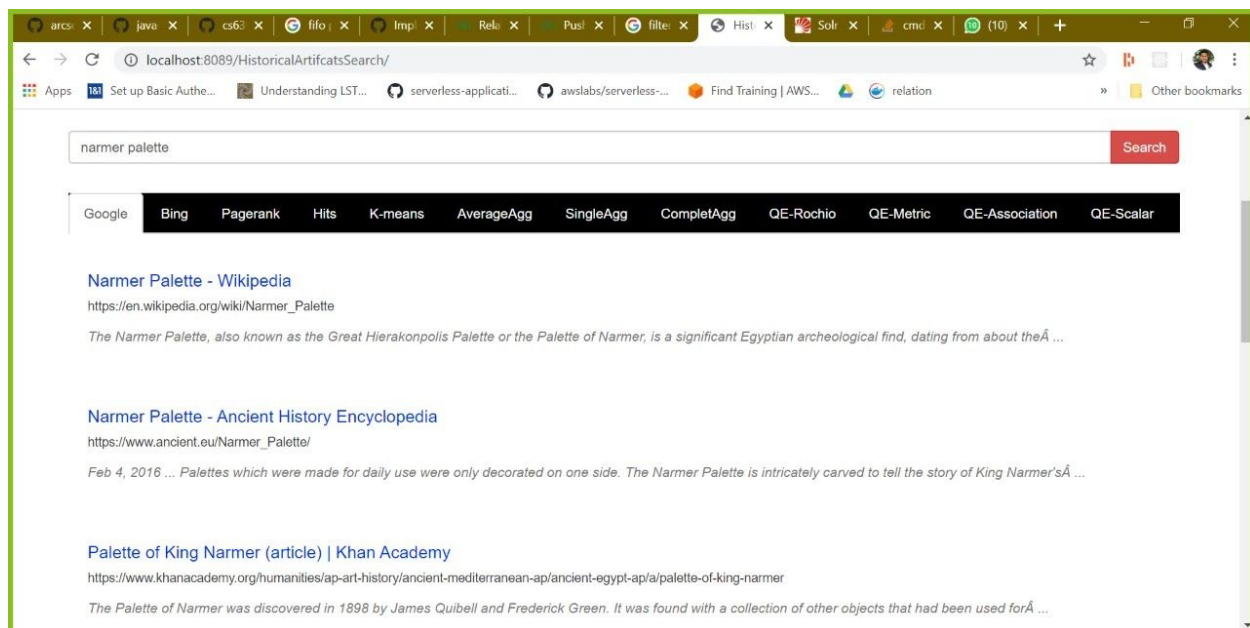
SCREENSHOTS OF RESULTS:

Home page for Search Engine



Search Results for Google

Query 1: Narmer Palette



Query 2: Pyramid

Google

Bing

Pagerank

Hits

K-means

AverageAgg

SingleAgg

CompletAgg

QE-Rochio

QE-Metric

QE-Association

QE-Scalar

[Pyramid Brewing Co.](#)
<https://www.pyramidbrew.com/>
The great beer we make today is a tribute to our rich brewing heritage of award- winning offerings, proudly brewed in the Pacific Northwest for over 30 years.

[Pyramid - Wikipedia](#)
<https://en.wikipedia.org/wiki/Pyramid>
*A pyramid (from Greek: πύραμις, *pyramís*) is a structure whose outer surfaces are ... For thousands of years, the largest structures on Earth were pyramidsâ€firstÂ ...*

[Welcome to Pyramid, a Python Web Framework](#)
<https://trypyramid.com/>
Pyramid is a lightweight Python web framework aimed at taking small web apps into big web apps. This site provides an easy entry point into Pyramid.

[Welcome to the Pylons Project](#)

Query 3: Sphinx of Hatshepsut

Google

Bing

Pagerank

Hits

K-means

AverageAgg

SingleAgg

CompletAgg

QE-Rochio

QE-Metric

QE-Association

QE-Scalar

[Sphinx of Hatshepsut | New Kingdom | The Met](#)
<https://www.metmuseum.org/en/art/collection/search/544442>
It was one of at least six granite sphinxes that stood in Hatshepsut's mortuary temple at Deir el-Bahri. The sphinx has a long history in Egyptian art, the most famousÂ ...

[Hatshepsut - Wikipedia](#)
<https://en.wikipedia.org/wiki/Hatshepsut>
Sphinx of Hatshepsut with unusual rounded ears and ruff that stress the lioness features of the statue, but with five toes â€newel post decorations from the lowerÂ ...

[Sphinx of Hatshepsut | Explore | MetKids | The Metropolitan Museum ...](#)
<https://www.metmuseum.org/art/online-features/metkids/explore/544442/Sphinx-of-Hatshepsut>
This sphinx was part of a set; at least six stood guard outside Hatshepsut's temple . How would you feel walking through that line of colossal sphinxes? Create.

[Sphinx of Hatshepsut - Female Pharaoh of Ancient Egypt â€ Joy of ...](#)

Search Results for Bing

Query 1: Narmer Palette

narmer palette

Google Bing Pagerank Hits K-means AverageAgg SingleAgg CompletAgg QE-Rochio QE-Metric QE-Association QE-Scalar

Narmer Palette - Wikipedia
https://en.wikipedia.org/wiki/Narmer_Palette
The Narmer Palette, also known as the Great Hierakonpolis Palette or the Palette of Narmer, is a significant Egyptian archeological find, dating from about the 31st century BC, belonging, at least nominally, to the category of Cosmetic palettes.

Narmer Palette - Ancient History Encyclopedia
https://www.ancient.eu/Narmer_Palette
The Narmer Palette (also known as Narmer's Victory Palette and the Great Hierakonpolis Palette) is an Egyptian ceremonial engraving, a little over two feet (64 cm) tall and shaped like a chevron shield, depicting the First Dynasty king Narmer conquering his enemies and uniting Upper and Lower Egypt. It features some of the earliest hieroglyphics found in Egypt and dates to c. 3200-3000 BCE.

Palette of King Narmer (article) | Khan Academy

Query 2: pyramid

pyramid

Google Bing Pagerank Hits K-means AverageAgg SingleAgg CompletAgg QE-Rochio QE-Metric QE-Association QE-Scalar

Pyramid Solitaire
<https://www.solitaire-pyramid.com>
Play a beautiful Pyramid solitaire game. No download necessary. Want More Solitaire Games? Try SolSuite Solitaire, the World's Most Complete Solitaire Collection with more than 700 solitaire games, 60 card sets, 300 card backs and 100 backgrounds! Try it now at www.solsuite.com

Pyramid - Wikipedia
<https://en.wikipedia.org/wiki/Pyramid>
A pyramid (from Greek: ἡ πυραμίδα, pyramís) is a structure whose outer surfaces are triangular and converge to a single step at the top, making the shape roughly a pyramid in the geometric sense. The base of a pyramid can be trilateral, quadrilateral, or of any polygon shape.

Egyptian Pyramids - Facts, Use & Construction - HISTORY
<https://www.history.com/topics/ancient-history/the-egyptian-pyramids>
The Pyramid of Khafre is the second tallest pyramid at Giza and contains Pharaoh Khafre's tomb. A unique feature built inside Khafre's pyramid complex was the Great Sphinx, a guardian statue ...

Query 3: Sphinx of Hatshepsut

Google

Bing

Pagerank

Hits

K-means

AverageAgg

SingleAgg

CompleatAgg

QE-Rochio

QE-Metric

QE-Association

QE-Scalar

[Sphinx of Hatshepsut | New Kingdom | The Met](#)
<https://www.metmuseum.org/art/collection/search/544442>

This colossal sphinx portrays the female pharaoh Hatshepsut with the body of a lion and a human head wearing a nemes headcloth and false beard. The sculptor has carefully observed the powerful muscles of the lion as contrasted to the handsome, idealized face of the pharaoh.

[Sphinx of Hatshepsut - Female Pharaoh of Ancient Egypt ...](#)
<https://joyofmuseums.com/.../highlights-of-the-met/the-sphinx-of-hatshepsut>

The Sphinx of the Foremost of Noble Ladies Hatshepsut means Foremost of Noble Ladies. She was one of only two female pharaohs in Ancient Egyptian history, who ruled as full Pharaoh not just as a regent for a younger male relative. She is the first significant female ruler in documented history. Born in 1507 BC, Hatshepsut [â€¦]

[Hatshepsut - Wikipedia](#)
<https://en.wikipedia.org/wiki/Hatshepsut>

Sphinx of Hatshepsut with unusual rounded ears and ruff that stress the lioness features of the statue, but with five toes and newel post decorations from the lower ramp of her tomb complex. The statue incorporated the nemes headcloth and a royal beard; two defining characteristics of an Egyptian pharaoh.

Search Results for PageRank

Query 1: Narmer Palette

arcs x java x cs63 x fifo x Imp x Relo x Pusi x filter x Hist x Solr x cmd x (10) x +

localhost:8089/HistoricalArtifactsSearch/

Apps Set up Basic Auth... Understanding LST... serverless-applicati... awslabs/serverless... Find Training | AWS... relation Other bookmarks

Google

Bing

Pagerank

Hits

K-means

AverageAgg

SingleAgg

CompleatAgg

QE-Rochio

QE-Metric

QE-Association

QE-Scalar

[Narmer - Wikipedia](#)
<https://en.wikipedia.org/wiki/Narmer>

Narmer - Wikipedia Narmer From Wikipedia, the free encyclopedia Jump to navigation Jump to search Ancient Egyptian pharaoh of the Early Dynastic Period Narmer Menes Verso of Narmer Palette Pharaoh Rei

[Narmer Palette - Ancient History Encyclopedia](#)
https://www.ancient.eu/Narmer_Palette/

Narmer Palette - Ancient History Encyclopedia Follow Us: Membership Encyclopedia Index Media Library Timeline Maps Random Weights & Measures Translations Audio Articles Education Teaching Materials Li

[Narmer Palette - Wikipedia](#)
https://en.wikipedia.org/wiki/Narmer_Palette

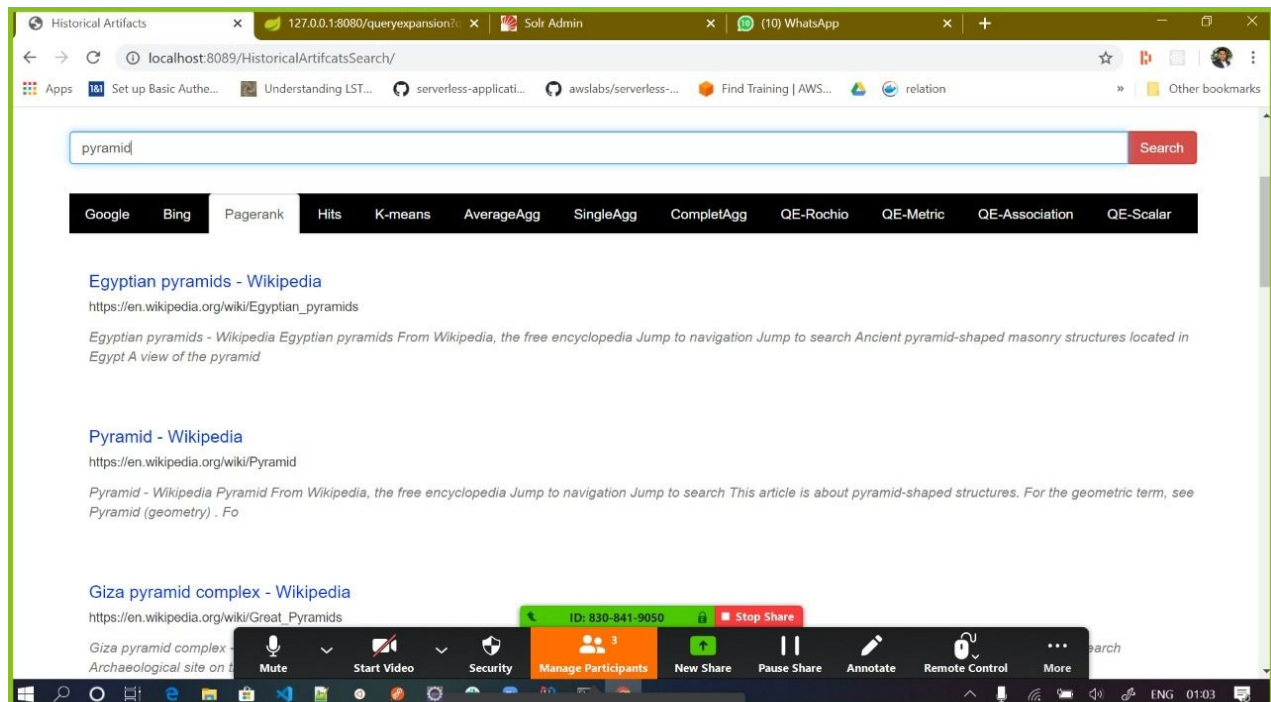
HITS.py ^ Lp5.zip ^ ID: 830-841-9050 Stop Share Show all x

Windows Taskbar

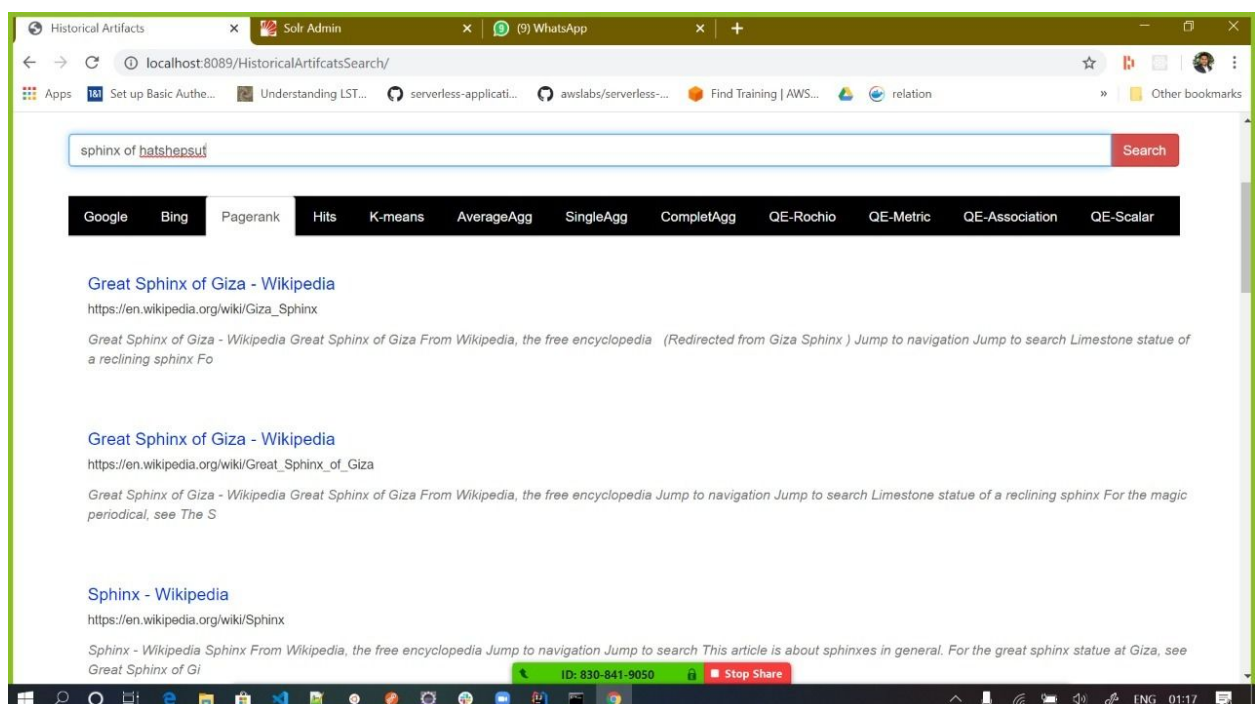
KRISHAN KUMAR

NETID: KXK180034

Query 2: pyramid



Query 3: Sphinx of Hatshepsut



Search Results for Hits:

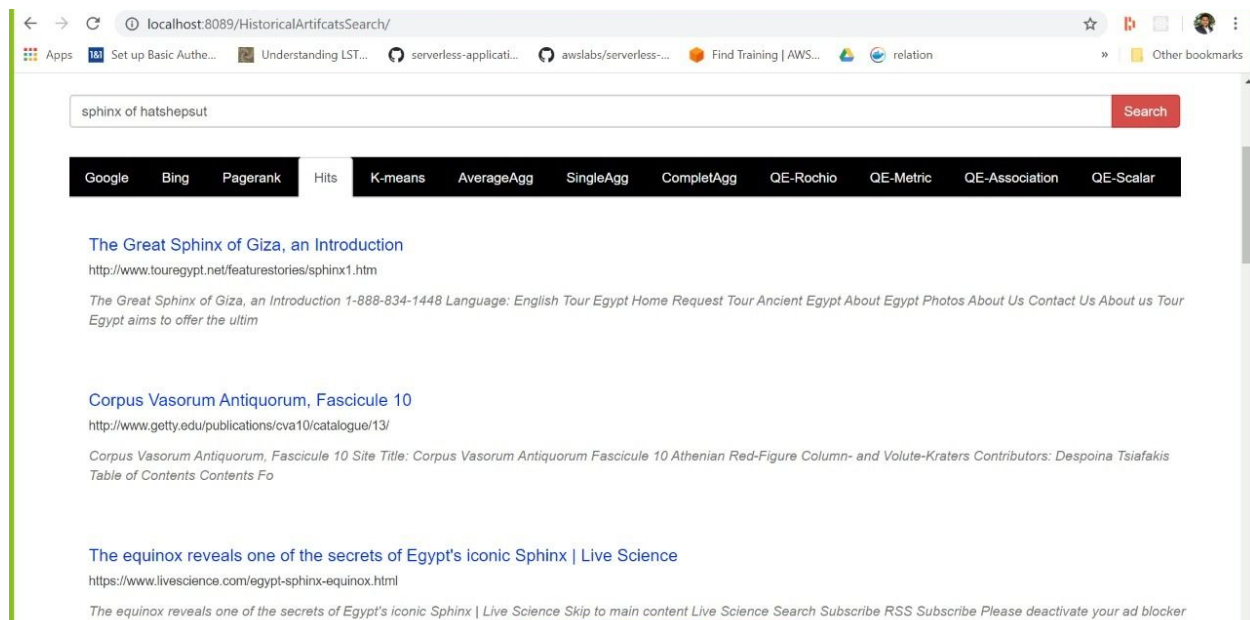
Query 1: Narmer Palette

The screenshot shows a web browser window with the address bar displaying 'localhost:8089/HistoricalArtifactsSearch/'. The search bar contains the text 'narmer palette' and a red 'Search' button. Below the search bar, there is a navigation bar with tabs for 'Google', 'Bing', 'Pagerank', 'Hits', 'K-means', 'AverageAgg', 'SingleAgg', 'CompleatAgg', 'QE-Rochio', 'QE-Metric', 'QE-Association', and 'QE-Scalar'. The 'Hits' tab is selected. The search results are displayed below the navigation bar. The first result is 'Narmer Palette - Ancient History Encyclopedia' with the URL 'https://www.ancient.eu/Narmer_Palette/'. The second result is 'Palette of King Narmer – Smarthistory' with the URL 'https://smarthistory.org/palette-of-king-narmer/'. The third result is 'Biography of Seth Peribsen | The Ancient Egypt Site' with the URL 'http://www.ancient-egypt.org/history/early-dynastic-period/2nd-dynasty/seth-peribsen/biography-of-seth-peribsen.html'. At the bottom of the browser window, there is a status bar showing 'HITS.py', 'Lp5.zip', and a green bar with the text 'ID: 830-841-9050' and a red 'Stop Share' button.

Query 2: pyramid

The screenshot shows a web browser window with the address bar displaying 'localhost:8089/HistoricalArtifactsSearch/'. The search bar contains the text 'pyramid' and a red 'Search' button. Below the search bar, there is a navigation bar with tabs for 'Google', 'Bing', 'Pagerank', 'Hits', 'K-means', 'AverageAgg', 'SingleAgg', 'CompleatAgg', 'QE-Rochio', 'QE-Metric', 'QE-Association', and 'QE-Scalar'. The 'Hits' tab is selected. The search results are displayed below the navigation bar. The first result is 'Egyptian pyramids - Wikipedia' with the URL 'https://en.wikipedia.org/wiki/Egyptian_pyramids'. The second result is 'The Great Pyramid of Giza (Pyramid of Cheops or Khufu) | Flickr' with the URL 'https://www.flickr.com/photos/jlascar/14823042753'. The third result is 'Teotihuacan - Wikipedia' with the URL 'https://en.wikipedia.org/wiki/Teotihuacan'. At the bottom of the browser window, there is a status bar showing 'ID: 830-841-9050' and a red 'Stop Share' button. The Windows taskbar is visible at the bottom of the screen, showing various application icons and the system clock.

Query 3: Sphinx of Hatshepsut

COLLABORATION WITH INDEXING PERSON (Arpita Dutta):

1. Solr is used for Page Ranking and indexing. The server for pagerank and indexing exposes an API for accessing the indexed results. The method creates the URL for a select query in Solr, and fetches the output for that query in Solr.
2. Calling the Solr server directly using the information provided by the student responsible for indexing and searching.
3. API also allows us to set some control parameters for e.g number of pages to be returned or format of the response. We have set the number of pages to be returned to be 15 and JSON format is used for the exchange of the data.
4. The python server API call returns results in json format for the PageRank and HITS algorithms. Information about this python API was collected from Arpita to make the API call. Since the PageRank and HITS algorithm is dependent on the indexing results from Solr, results from Solr API were sent to this API via POST and the response was collected back.

The following fields are selected for display in the UI:

URL: the URL address of the page

Title: Title of the page.

Description: HTML content of the page (with HTML tags removed) – truncated to 200 characters for display in the UI as a short description about the webpage.

COLLABORATION WITH CLUSTERING PERSON (Pavan Vaidya):

1. Clustering algorithms like K-means and agglomerative clustering have been implemented as a POST Call on a separate server. This JAVA API call information is collected from Pavan and then the results are rendered on the UI as a json object.
2. The top results from solr are passed to the cluster based sorting algorithm. The most relevant cluster is identified and the documents which belong to the same cluster are retrieved. Top 12 results in the UI are shown, based on the chosen cluster.

COLLABORATION WITH QUERY EXPANSION PERSON (Arihant Chhajed):

1. The query expansion algorithm is implemented and hosted in a separate server. The input to the Query expansion API is the initial query string made by the user and the type of query expansion algorithm to be used.
2. Final output from the API is the expanded query string. This expanded string is then used to retrieve a new set of results from the Solr API.
3. These are rendered on the UI as a json object.

DISCUSSION:

1. After that we discussed how we will be setting up the UI and what languages would be easier to integrate the clustering, indexing and query expansion part with the UI. We had multiple discussions about how the expanded query and result should be displayed on the UI, also how to customize search variables/tabs, how to implement pagerank, HITS in combination with clustering and query expansion.
2. We did daily Zoom/Microsoft Teams meetings, sharing screens for quick communication and checking the workflow of the project as well as what needs to be worked on. For quick progress we also used whatsapp group to communicate.

CONCLUSION:

The initial UI was quite plain and simple, as we got to the enhancement part, we were able to add a few more modifications to it to make it look better and incorporate all the tabs for clustering and query expansion.

CLUSTERING

1. Performed K-means, single link, complete link and average link types of clustering.
(ie flat and agglomerative clustering methods)

FLAT CLUSTERING (K-MEANS):

- *k*-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in Information Retrieval. *k*-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.
- To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids
- It halts creating and optimizing clusters when either:
 1. The centroids have stabilized — there is no change in their values because the clustering has been successful.
 2. The defined number of iterations has been achieved.

HIERARCHICAL CLUSTERING (AGGLOMERATIVE):

- Hierarchical clustering treats each data point as a singleton cluster, and then successively merges clusters until all points have been merged into a single remaining cluster. A hierarchical clustering is often represented as a dendrogram.
- In complete-link (or complete linkage) hierarchical clustering, we merge in each step the two clusters whose merger has the smallest diameter (or: the two clusters with the smallest **maximum** pairwise distance).
- In single-link (or single linkage) hierarchical clustering, we merge in each step the two clusters whose two closest members have the smallest distance (or: the two clusters with the smallest **minimum** pairwise distance).
- Average link (or average linkage) is a compromise between the sensitivity of complete-link clustering to outliers and the tendency of single-link clustering to form long chains that do not correspond to the intuitive notion of clusters as compact, spherical objects.

IMPLEMENTATION:

How the clustering results are displayed onto the UI:

1. A JAVA servlet called CLusterServlet.java has been implemented with all the clustering codes.
2. The UI makes an API call to the clustering service when the user selects a particular clustering type in the search query at <http://localhost:5001/geCluser/clustertype> .
3. The request as POST type and the body of the servlet code has the k-means clustering and agglomerative clustering - single, average, complete algorithms implemented.
4. When the user enters the search query along with the clustering type, the documents are fetched according to the search query from the Solr service API and forwarded to this clustering service.
5. In this service, according to the type of clustering algorithm chosen, the clustering is applied on the input documents and then the clustered results are rendered back onto the UI as a json object.
6. For the clustering algorithms, the built-in sklearn library is used for k means and agglomerative clustering.
7. K = 12 which is the optimum k value for our data and index, is used for both types of clustering methods.

Choosing the value of K

Determining the **optimal number of clusters** in a data set is a fundamental issue in partitioning clustering, such as k-means clustering, which requires the user to specify the number of clusters k to be generated. There is no definitive answer to this question. The optimal number of clusters is somehow subjective and depends on the method used for measuring similarities and the parameters used for partitioning. A simple and popular solution consists of inspecting the dendrogram produced using hierarchical clustering to see if it suggests a particular number of clusters. This approach is also subjective.

I used silhouette Coefficient to determine the value of K. The Silhouette Coefficient is calculated using the mean intra-cluster distance and the mean nearest cluster distance for each sample. The Silhouette Coefficient for a sample is . To clarify, b is the distance between a sample and the nearest cluster that the sample is not a part of. Note that Silhouette Coefficient is only defined if the number of labels is $2 \leq n_labels \leq n_samples - 1$.

The best value is 1 and the worst value is -1. Values near 0 indicate overlapping clusters. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.

I generated the silhouette coefficients from 10 to 15 and 16 to 21 and found that the k value of 12 gives the best result. Hence I chose k as 12

State clearly how many queries you have used to test the impact of the results of each clustering method, how you have generated them and how you have judged the results of your relevance models :

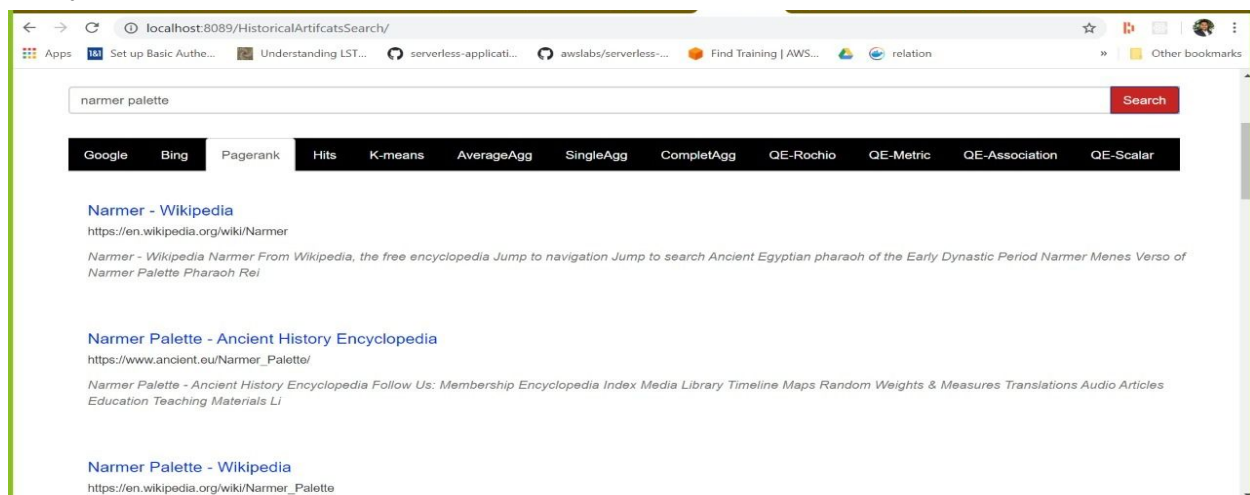
I have used 20 queries to test the impact of the results of K Means clustering method. The queries were generated keeping the wide diversity of historical artifacts in mind. The queries encompass a wide range of topics related to historical artifacts like egyptian, greek, asian, roman as well as other cultural artifacts. If the urls are a match they would show the result on the UI. The results were judged by the url and url content which indicated if the results were relevant to the query or not.

Discuss how you have decided to select the queries for the demonstration of your search engine. Provide three examples of the queries and the results produced by your search engine and the clusters that you have created.

For demonstration purposes I have decided to select queries from the test cases which have given relevant results.

Original Search Results from PageRank:

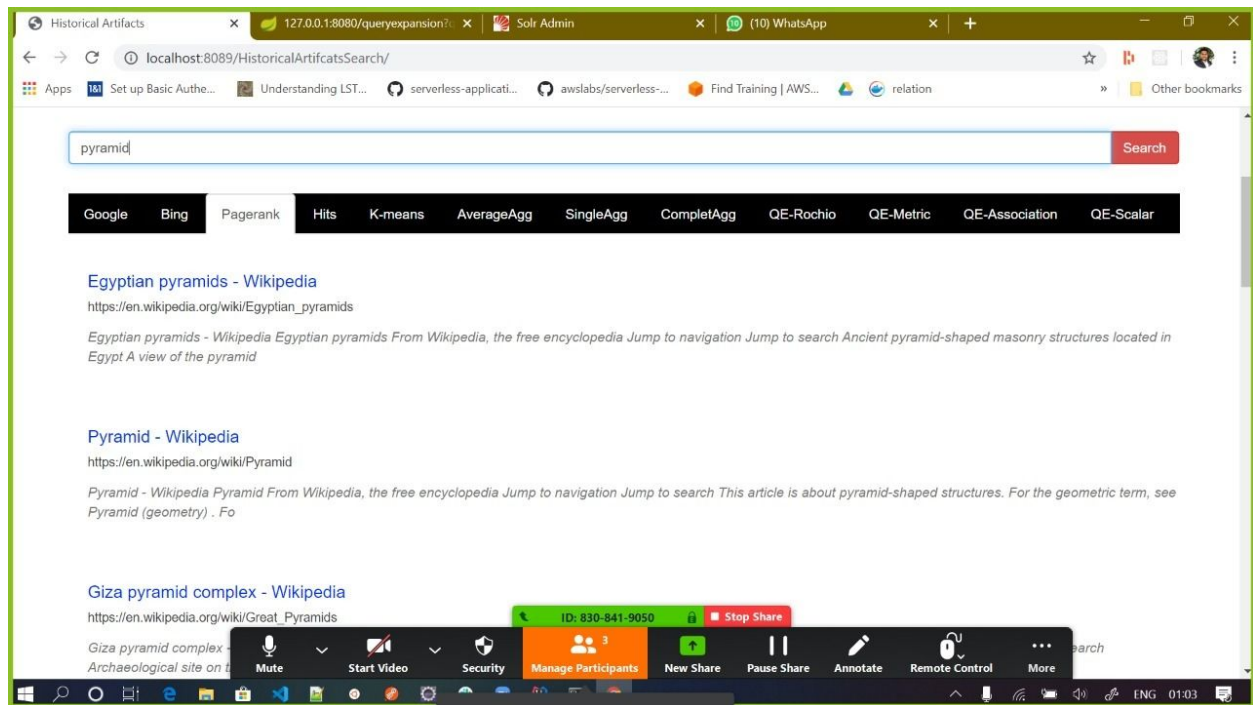
Query 1: Narmer Palette



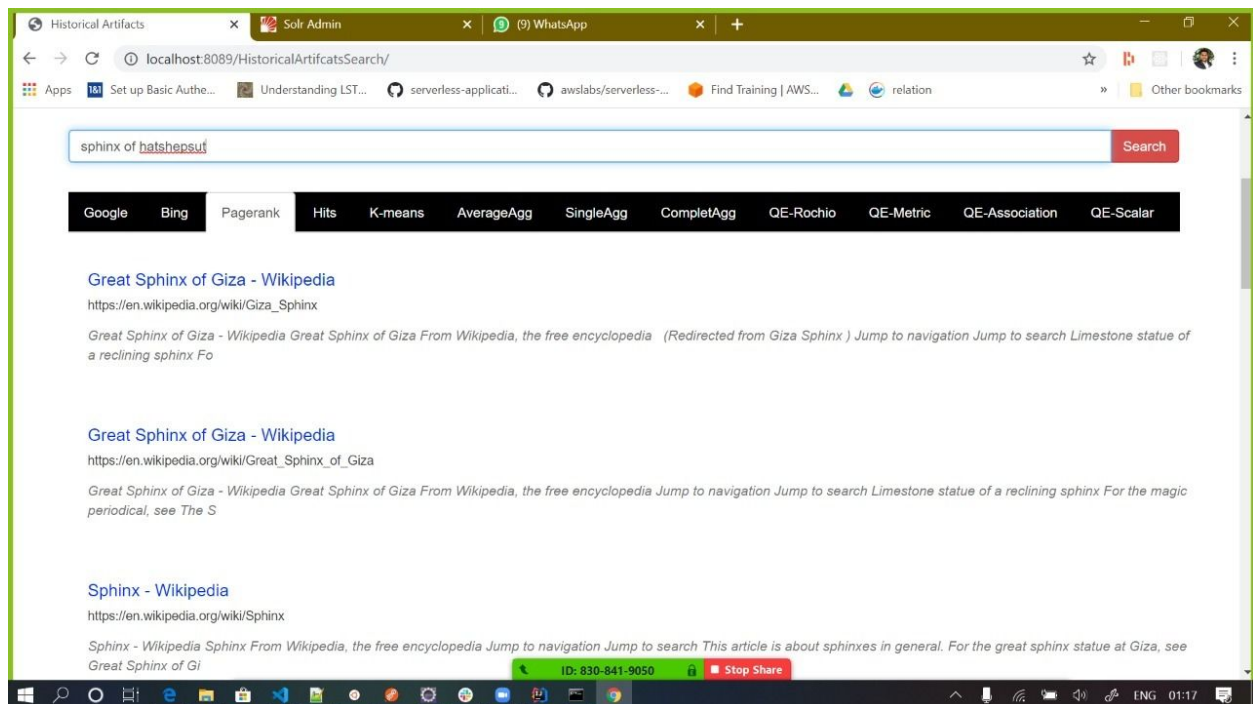
PAWAN VAIDYA

NETID: PAV180001

Query 2: pyramid



Query 3: sphinx of Hatshepsut



PAWAN VAIDYA

NETID: PAV180001

Search Results from K-Means Clustering

Query 1: Narmer Palette

The screenshot shows a web browser window with the URL `localhost:8089/HistoricalArtifactsSearch/`. The search bar contains the text "narmer palette". Below the search bar, there is a navigation bar with tabs: Google, Bing, Pagerank, Hits, K-means (selected), AverageAgg, SingleAgg, CompletAgg, QE-Rochio, QE-Metric, QE-Association, and QE-Scalar. A "Cluster Chart" button is visible. The search results are displayed in a list format, showing the top three results:

- Upper Egypt - Wikipedia**
https://en.wikipedia.org/wiki/Upper_Egypt
Upper Egypt - Wikipedia Upper Egypt From Wikipedia, the free encyclopedia Jump to navigation Jump to search This article's lead section does not adequately summarize key points of its contents . Pleas
- Abydos, Egypt - Wikipedia**
https://en.wikipedia.org/wiki/Abydos,_Egypt
Abydos, Egypt - Wikipedia Abydos, Egypt From Wikipedia, the free encyclopedia Jump to navigation Jump to search For other uses, see Abydos . City in ancient Egypt Abydos
أبيدوس Façade of the Temple of
- 4th millennium BC - Wikipedia**

The bottom of the screen shows a Windows taskbar with various icons and a system tray displaying the time as 23:46.

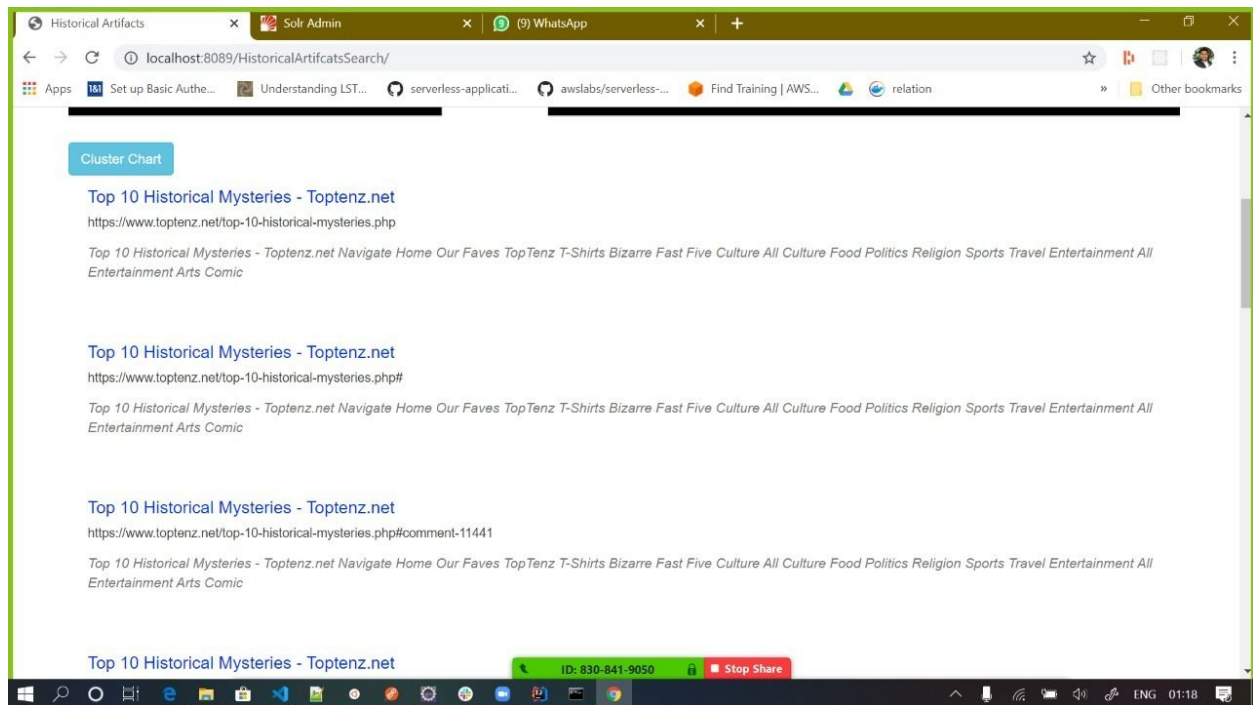
Query 2: pyramid

The screenshot shows the same web browser window with the URL `localhost:8089/HistoricalArtifactsSearch/`. The search bar now contains the text "pyramid". The navigation bar remains the same, with the "K-means" tab selected. The search results are displayed in a list format, showing the top three results:

- A Museum for Everyone The Louvre Pyramid Turns 30 | Louvre Museum | Paris**
<https://www.louvre.fr/en/museum-everyonethe-louvre-pyramid-turns-30>
A Museum for Everyone The Louvre Pyramid Turns 30 | Louvre Museum | Paris Go to content Go to navigation Go to search Change language Accessibility Teachers Professionals & Associations Sign in Crea
- A Museum for Everyone The Louvre Pyramid Turns 30 | Louvre Museum | Paris**
<https://www.louvre.fr/en/museum-everyonethe-louvre-pyramid-turns-30#>
A Museum for Everyone The Louvre Pyramid Turns 30 | Louvre Museum | Paris Go to content Go to navigation Go to search Change language Accessibility Teachers Professionals & Associations Sign in Crea
- The Origins of the**

The bottom of the screen shows a Windows taskbar with various icons and a system tray displaying the time as 01:04.

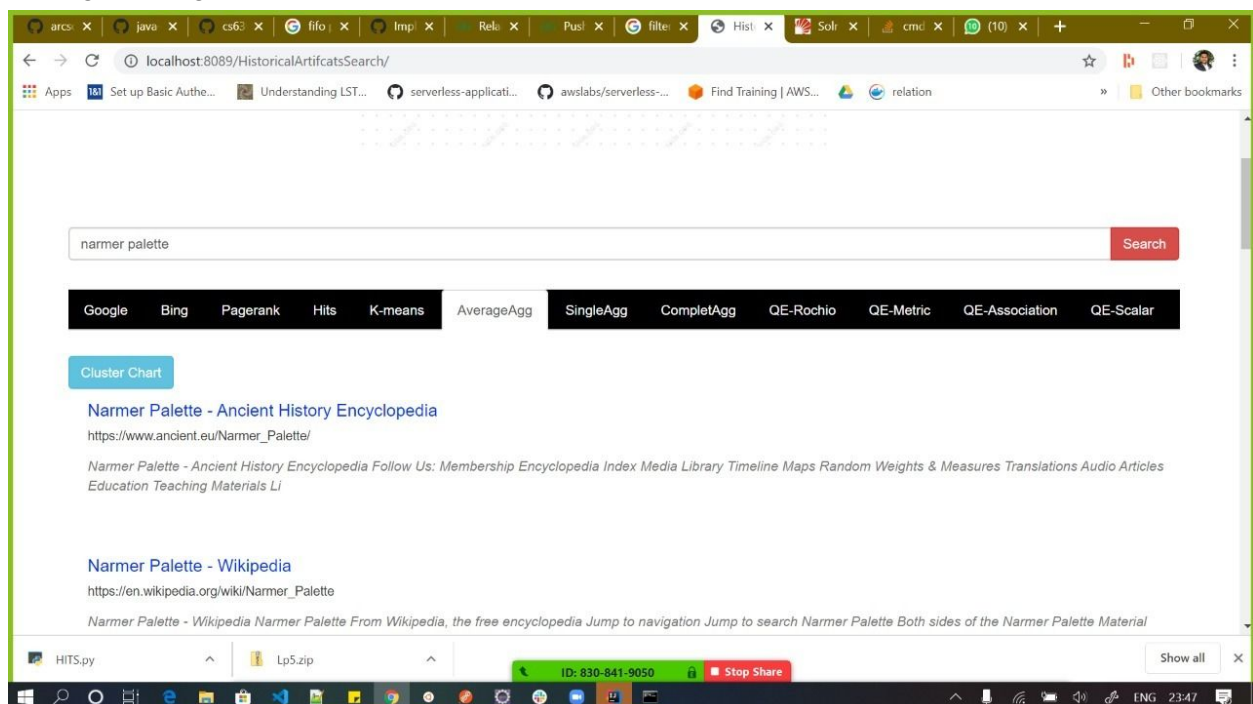
Query 3: Sphinx of Hatshepsut



Search Results from Agglomerative Clustering

Query 1: Narmer Palette

Average Linkage



PAWAN VAIDYA

NETID: PAV180001

Single Linkage

The screenshot shows a web browser window with the URL `localhost:8089/HistoricalArtifactsSearch/`. The search bar contains the text "narmer palette". Below the search bar, there is a row of buttons for different clustering methods: Google, Bing, Pagerank, Hits, K-means, AverageAgg, SingleAgg, CompletAgg, QE-Rochio, QE-Metric, QE-Association, and QE-Scalar. The "SingleAgg" button is highlighted. Below this row, there is a "Cluster Chart" button. The search results are displayed as a list of links:

- Narmer - Wikipedia**
<https://en.wikipedia.org/wiki/Narmer>
Narmer - Wikipedia Narmer From Wikipedia, the free encyclopedia Jump to navigation Jump to search Ancient Egyptian pharaoh of the Early Dynastic Period Narmer Menes Verso of Narmer Palette Pharaoh Rei
- Narmer Palette - Ancient History Encyclopedia**
https://www.ancient.eu/Narmer_Palette/
Narmer Palette - Ancient History Encyclopedia Follow Us: Membership Encyclopedia Index Media Library Timeline Maps Random Weights & Measures Translations Audio Articles Education Teaching Materials Li
- Narmer Palette - Wikipedia**
https://en.wikipedia.org/wiki/Narmer_Palette

The bottom of the browser window shows the Windows taskbar with various icons and a system clock displaying 23:47.

Complete Linkage

The screenshot shows the same web browser window as the previous one, but with the "CompletAgg" button highlighted in the row of clustering methods. The search results are identical to the previous screenshot:

- Narmer - Wikipedia**
<https://en.wikipedia.org/wiki/Narmer>
Narmer - Wikipedia Narmer From Wikipedia, the free encyclopedia Jump to navigation Jump to search Ancient Egyptian pharaoh of the Early Dynastic Period Narmer Menes Verso of Narmer Palette Pharaoh Rei
- Narmer Palette - Ancient History Encyclopedia**
https://www.ancient.eu/Narmer_Palette/
Narmer Palette - Ancient History Encyclopedia Follow Us: Membership Encyclopedia Index Media Library Timeline Maps Random Weights & Measures Translations Audio Articles Education Teaching Materials Li
- Narmer Palette - Wikipedia**
https://en.wikipedia.org/wiki/Narmer_Palette

The bottom of the browser window shows the Windows taskbar with various icons and a system clock displaying 23:47.

PAWAN VAIDYA

NETID: PAV180001

Query 2: pyramid Average linkage

The screenshot shows a web browser window with the URL `localhost:8089/HistoricalArtifactsSearch/`. The search bar contains the text "pyramid". Below the search bar, there is a navigation menu with the following options: Google, Bing, Pagerank, Hits, K-means, AverageAgg (selected), SingleAgg, CompletAgg, QE-Rochio, QE-Metric, QE-Association, and QE-Scalar. A "Cluster Chart" button is visible on the left. The search results are displayed as a list of links from Wikipedia:

- Egyptian pyramids - Wikipedia**
https://en.wikipedia.org/wiki/Egyptian_pyramids
Egyptian pyramids - Wikipedia Egyptian pyramids From Wikipedia, the free encyclopedia Jump to navigation Jump to search Ancient pyramid-shaped masonry structures located in Egypt A view of the pyramid
- Pyramid - Wikipedia**
<https://en.wikipedia.org/wiki/Pyramid>
Pyramid - Wikipedia Pyramid From Wikipedia, the free encyclopedia Jump to navigation Jump to search This article is about pyramid-shaped structures. For the geometric term, see Pyramid (geometry) . Fo
- Giza pyramid complex - Wikipedia**
https://en.wikipedia.org/wiki/Giza_pyramid_complex

The bottom of the screenshot shows a Windows taskbar with various icons and a system tray displaying the time as 01:04.

Complete linkage

The screenshot shows a web browser window with the URL `localhost:8089/HistoricalArtifactsSearch/`. The search bar contains the text "pyramid". Below the search bar, there is a navigation menu with the following options: Google, Bing, Pagerank, Hits, K-means, AverageAgg, SingleAgg, CompletAgg (selected), QE-Rochio, QE-Metric, QE-Association, and QE-Scalar. A "Cluster Chart" button is visible on the left. The search results are displayed as a list of links from Britannica:

- Pyramid | architecture | Britannica**
<https://www.britannica.com/technology/pyramid-architecture>
Pyramid | architecture | Britannica Search Britannica Encyclopædia Britannica Login Subscribe Now Categories Science Technology Health & Medicine Sports & Recreation Geography & Travel World History P
- Pyramids of Giza | History & Facts | Britannica**
<https://www.britannica.com/topic/Pyramids-of-Giza>
Pyramids of Giza | History & Facts | Britannica Search Britannica Encyclopædia Britannica Login Subscribe Now Categories Science Technology Health & Medicine Sports & Recreation Geography & Travel Wor
- Şaqqārah | archaeological site, Memphis, Egypt | Britannica**
<https://www.britannica.com/place/Saqqarah>

Single linkage

pyramid Search

Google Bing Pagerank Hits K-means AverageAgg SingleAgg CompletAgg QE-Rochio QE-Metric QE-Association QE-Scalar

Cluster Chart

Egyptian pyramids - Wikipedia
https://en.wikipedia.org/wiki/Egyptian_pyramids
Egyptian pyramids - Wikipedia Egyptian pyramids From Wikipedia, the free encyclopedia Jump to navigation Jump to search Ancient pyramid-shaped masonry structures located in Egypt A view of the pyramid

Pyramid - Wikipedia
<https://en.wikipedia.org/wiki/Pyramid>
Pyramid - Wikipedia Pyramid From Wikipedia, the free encyclopedia Jump to navigation Jump to search This article is about pyramid-shaped structures. For the geometric term, see Pyramid (geometry) . Fo

Giza pyramid complex - Wikipedia
https://en.wikipedia.org/wiki/Great_Pyramids
Giza pyramid complex - Wikipedia Giza pyramid complex From Wikipedia, the free encyclopedia (Redirected from Great Pyramids) Jump to navigation Jump to search

Query 3: Sphinx of Hatshepsut
Average Linkage

Sphinx of Hatshepsut

Google Bing Pagerank Hits K-means AverageAgg SingleAgg CompletAgg QE-Rochio QE-Metric QE-Association QE-Scalar

Cluster Chart

Top 10 Historical Mysteries - Toptenz.net
<https://www.toptenz.net/top-10-historical-mysteries.php>
Top 10 Historical Mysteries - Toptenz.net Navigate Home Our Faves TopTenz T-Shirts Bizarre Fast Five Culture All Culture Food Politics Religion Sports Travel Entertainment All Entertainment Arts Comic

Top 10 Historical Mysteries - Toptenz.net
<https://www.toptenz.net/top-10-historical-mysteries.php#>
Top 10 Historical Mysteries - Toptenz.net Navigate Home Our Faves TopTenz T-Shirts Bizarre Fast Five Culture All Culture Food Politics Religion Sports Travel Entertainment All Entertainment Arts Comic

Top 10 Historical Mysteries - Toptenz.net
<https://www.toptenz.net/top-10-historical-mysteries.php#comment-11441>
Top 10 Historical Mysteries - Toptenz.net Navigate Home Our Faves TopTenz T-Shirts Bizarre Fast Five Culture All Culture Food Politics Religion Sports Travel Entertainment All Entertainment Arts Comic

PAWAN VAIDYA

NETID: PAV180001

Complete Linkage

The screenshot shows a web browser window with the address bar displaying 'localhost:8089/HistoricalArtifactsSearch/'. The browser has several tabs open, including 'Historical Artifacts', 'Solr Admin', and '(9) WhatsApp'. The 'Solr Admin' tab is active, showing a search results page. The page has a navigation bar with tabs: 'Google', 'Bing', 'Pagerank', 'Hits', 'K-means', 'AverageAgg', 'SingleAgg', 'CompleteAgg', 'QE-Rochio', 'QE-Metric', 'QE-Association', and 'QE-Scalar'. The 'CompleteAgg' tab is selected. Below the navigation bar, there is a 'Cluster Chart' button. The search results are displayed in a list format, showing the title 'Great Sphinx of Giza - Wikipedia', the URL 'https://en.wikipedia.org/wiki/Giza_Sphinx', and a snippet of text: 'Great Sphinx of Giza - Wikipedia Great Sphinx of Giza From Wikipedia, the free encyclopedia (Redirected from Giza Sphinx) Jump to navigation Jump to search Limestone statue of a reclining sphinx Fo'. The results are sorted by relevance, with the most relevant result at the top.

Single Linkage

The screenshot shows a web browser window with the address bar displaying 'localhost:8089/HistoricalArtifactsSearch/'. The browser has several tabs open, including 'Historical Artifacts', 'Solr Admin', and '(9) WhatsApp'. The 'Solr Admin' tab is active, showing a search results page. The page has a navigation bar with tabs: 'Google', 'Bing', 'Pagerank', 'Hits', 'K-means', 'AverageAgg', 'SingleAgg', 'CompleteAgg', 'QE-Rochio', 'QE-Metric', 'QE-Association', and 'QE-Scalar'. The 'SingleAgg' tab is selected. Below the navigation bar, there is a 'Cluster Chart' button. The search results are displayed in a list format, showing the title 'Great Sphinx of Giza - Wikipedia', the URL 'https://en.wikipedia.org/wiki/Giza_Sphinx', and a snippet of text: 'Great Sphinx of Giza - Wikipedia Great Sphinx of Giza From Wikipedia, the free encyclopedia (Redirected from Giza Sphinx) Jump to navigation Jump to search Limestone statue of a reclining sphinx Fo'. The results are sorted by relevance, with the most relevant result at the top.

COLLABORATION WITH THE UI PERSON (Krishan Duhan):

1. Clustering algorithms like K-means and agglomerative clustering have been implemented as a POST request on a separate local server. This JAVA API call information is sent to Krishan and then the results are rendered on the UI.

The top results from solr service are passed to the cluster based sorting algorithm. The most relevant cluster is identified and the documents which belong to the same cluster are retrieved. Top 12 results in the UI are shown, based on the chosen cluster and displayed on the UI.

COLLABORATION WITH THE INDEXING PERSON (Arpita Dutta):

1. Solr service is used for Page Ranking and indexing. The server for pagerank and indexing exposes an API for accessing the indexed results. The method creates the URL for a select query in Solr, and fetches the output for that query in Solr.
2. Calling the Solr server directly using the information provided by the student responsible for indexing and searching (Arpita).
3. API also allows us to set some control parameters for e.g number of pages to be returned or format of the response. We have set the number of pages to be returned to be 15 and JSON format is used for the exchange of the data.
4. Using the Solr service, Pagerank and HITS python service and the clustering service, altogether the url results are combined according to user search query and clustering type and displayed on the UI as output.

DISCUSSION:

The clustering codes written initially were to be modified to be programmed as API endpoints and we had to collaborate to make the results can be rendered in the UI efficiently.

CONCLUSION:

Clustering is a very useful technique to provide relevant page results. From the queries tested we were able to find that the clustering and query expansion results were much more relevant than the initial ones displayed. Hence it was helpful to perform these last two steps so as to enhance our search engine. It is also heavily dependent on the initial result set of documents, hence, some improvement can be done to reduce that dependency.

QUERY EXPANSION AND RELEVANCE FEEDBACK

For Query expansion, 4 methods were used for expanding the search query for relevant results.

1. Rocchio Algorithm
2. Association cluster
3. Metric cluster
4. Scalar cluster


ROCCHIO ALGORITHM:

For Rocchio Algorithm, Inspiration was taken from the Lucene query expansion module. The idea behind the concept is to expand query using the following method

Rocchio 1971 Algorithm (SMART)

- Used in practice:

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

- D_r = set of known relevant doc vectors
- D_{nr} = set of known irrelevant doc vectors
 - Different from C_r and C_{nr} 
- \vec{q}_m = modified query vector; \vec{q}_0 = original query vector; α, β, γ : weights (hand-chosen or set empirically)
- New query moves toward relevant documents and away from irrelevant documents

The values of alpha and beta were taken to be 1 and 0.75. The decay factor was not taken into account as the query expansion module of lucene disregards that. Still, the query expansion worked well for most of the queries.

The constants alpha and beta were then used to boost the query and the relevant document terms and again adjusted and queried to find more relevant results.

The following table is the result of 20 tested queries through the Rocchio algorithm and the relevant values corresponding to it.

The 20 queries were selected to test out the rocchio giving relevant expanded queries to the topic “ ”. Later, I have discussed the search results and the observations made.

Sno.	Query Term	Expanded Query Term	Number of relevant documents	Number of Irrelevant documents
1	Pyramid	Giza, Egypt	10	5
2.	Museum	historical, world, collections	15	3
3	Sphinx	Giza, Hatshepsut	8	5
4	Narmer Palette	Egypt, king	12	3
5	Agamemnon Mask	King, Greece, Troy	18	4
6	Terracotta figure	Asian, Buddhist, Indus valley, Alexandar, emperor	20	11
7	Dish	Carved, islamic design, Peru, Spanish	35	12
8	Bowl	Carved, clay	40	10
9	Carpet	Islamic, royal, import	44	6
10	Hand Axe	Neanderthal, egypt sand axe	3	1
11	Nefertiti figure	Egypt, queen, aten	10	4
12	Akhenaten figure	King, aten,	14	7
13	sarcophagus	Mummy, coffin, drawing	34	24
14	Alexander sculpture	Macedonia, king, emperor, the great	23	10
15	Buddha	Figure, Nepal, India, Korea	27	17
16	Achilles	Troy, Greece, foot,	48	13

		arrow		
ARIHANT CHHAJED			NETID: ARC180006	
17	Priam treasure	Necklace, king, troy, hector	30	4
18	Royal earrings	Indian, greek, egyptian	15	2
19	jar	Greek, story , urn, wine	34	11
20	coin	Currency, macedonian, indus, trade	20	2

Observations of the above result:

Rocchio algorithm works well with most queries and expands on relevant terms, but some query results were highly skewed as the crawled data was topic specific.

Changing the alpha and beta values also helped in some cases.

ASSOCIATIVE, METRIC AND SCALAR CLUSTERING:

Table of 50 queries used for relevance feedback:

Sr. No	Query	Association Expanded Query (new added words)	Metric Expanded Query (new added words)	Scalar Expanded Query (new added words)
1	Pyramid	Giza, Egypt,dynasty,greate,,khufu,..	Giza	Ancient,, Egypt
2.	Museum	famous , world	famous	collections
3	Sphinx	Giza,triangle,demisch	Hatshepsut	Egypt,cultural,hona rhaye
4	Narmer Palette	Egypt	king	Egypt

ARIHANT CHHAJED				NETID: ARC180006
5	Agamemnon Mask	King, Greece, Troy	Greece, Troy	Greece
6	Terracotta figure	Asian	Asian, Buddhist	Indus valley
7	Dish	Carved	islamic design	food
8	Bowl	decoration	clay	Carved
9	Carpet	Islamic	royal, import	import,decor
10	Hand Axe	Neanderthal	egypt, sand axe	hunt
11	Nefertiti figure	Egypt, queen, aten	queen	Egypt
12	Akhenaten figure	King, aten,	King, aten,	King, aten,
13	sarcophagus	Mummy	coffin	drawing
14	Alexander sculpture	Macedonia, king, emperor, the great	great	king, emperor
15	Buddha	Figure, Nepal, India, Korea	India	Figure, Nepal
16	Achilles	Troy, Greece	foot, arrow	foot
17	Priam treasure	Necklace	troy, hector	Necklace, king, troy, hector
18	Royal earrings	Indian	Indian, greek	egyptian
19	jar	Greek, story	urn, wine	urn, wine
20	coin	Currency, macedonian, indus, trade	macedonian, indus, trade	Currency, indus, trade
21	Aristocrates Bust	Greek	sculpture	scholar
22	statuete	Figurine, goddess	Asian, Egyptian	tomb

ARIHANT CHHAJED				NETID: ARC180006
23	Tripod Jar	ebony	glass	jar
24	Neolithic Painted Pottery	urn	Pottery, clay	neolithic
25	Mohenjo-daro	civilization	Indus Valley	Indian ancient
26	artifacts	historical	war	Scrolls, paintings
27	Law Code of Hammurabi	code	Hammurabi	Book, babylon
28	Tutankhamun tomb	egypt	Tomb, sarcophagus	King, nefertiti
29	Teracotta army	chinese	Army, soldiers	massacre
30	pompeii	volcano	Burnt city	Games, senator, rome
31	peking man	china	Homo erectus	fossil
32	Dead sea scrolls	Jewish	manuscripts	Judean desert
33	Tomb of Philip	Macedonia	Alexander	king
34	drinking cup	wine	Greek, rome	Bronze, ivory
35	Amun	God	Egypt	Pharoah
36	amulet	Charm, latin	Good luck	Blessings, egypt
37	Textile fragment	carpet	Islamic design	civilization history
38	Carpet	textile	islamic	artwork
39	Ishtar Gate	babylon	Entry gate	Pergamon museum
40	The Temple of Dendur	Egypt	Petronius	Augustus, Nubian
41	The Pergamon	Berlin	King Eumenes	Greek acropolis

	Altar			
ARIHANT CHHAJED			NETID: ARC180006	
42	The Rosetta Stone	Memphis, egypt	King Ptolemy	Granite stone
43	The Winged Victory of Samothrace	Hellenistic sculpture	Louvre display	Nike of Samothrace
44	The Parthenon Marbles	Greek marble	sculpture	Phidias architect
45	Babylon scrolls	Babylon	Egypt	papyrus
46	Kleroterion	democracy	randomization	tool
47	Antikythera Mechanism	Greek computer	Hand powered	astronomical
48	Burmese Bronze dragon	burmeze	dragon	Bronze canon
49	Sword of Emperor Maximilian	sword	Emperor Maximilian	Austria
50	Chinese Qin Sword	China	Qin Dynasty	Sword

Observations:

Metric cluster performed the best along with associative cluster and scalar cluster performing similarly (scalar being slightly better than associative). This may be due to the fact that the “neighbouring terms” play an important role in query expansion and bring out more relevant queries.

Associative clustering:

Query 1: Narmer Palette

Resulting expanded queries and search results

The screenshot shows a web browser window with the URL `localhost:8089/HistoricalArtifactsSearch/`. The page features a logo for "HISTORICAL ARTIFACTS" with the tagline "Your personal search engine". Below the logo is a search bar containing the text "narmer palette" and a red "Search" button. Underneath the search bar is a horizontal menu with various search engines and algorithms: Google, Bing, Pagerank, Hits, K-means, AverageAgg, SingleAgg, CompletAgg, QE-Rochio, QE-Metric, QE-Association, and QE-Scalar. The main content area displays the text "Showing results for - narmer,palette,New,beginning,fig,sites,Connor,den,An,catfish,Archaeology,Iry,des,identified,enclosure,presence,7". Below this, there is a link to "Narmer - Wikipedia" with the URL `https://en.wikipedia.org/wiki/Narmer`.

Query 2:pyramid

Resulting expanded queries and search results

The screenshot shows the same web browser window as the previous one, but with the search bar containing the text "pyramid". The horizontal menu remains the same. The main content area displays the text "Showing results for - pyramid,Pyramids,BC,built,Egypt,Giza,pyramids,Dynasty,article,Great,Pyramid,Egyptian,Ancient,Khufu,complex,f". Below this, there are three links to Wikipedia articles: "Giza pyramid complex - Wikipedia" with the URL `https://en.wikipedia.org/wiki/Great_Pyramids`, "Egyptian pyramids - Wikipedia" with the URL `https://en.wikipedia.org/wiki/Egyptian_pyramids`, and "Great Pyramid of Giza - Wikipedia".

Query 3: Sphinx of Hatshepsut

Resulting expanded queries and search results

Showing results for - *Sphinx,of,Hatshepsut,city,Triangle,Demisch,Pausanias,decorations,Fugiens,Often,figurine,Lanka,vocabulary,zur,v*

[Sphinx - Wikipedia](#)
<https://en.wikipedia.org/wiki/Sphinx>
Sphinx - Wikipedia Sphinx From Wikipedia, the free encyclopedia Jump to navigation Jump to search This article is about sphinxes in general. For the great sphinx statue at Giza, see Great Sphinx of Giza

[Mut - Wikipedia](#)
<https://en.wikipedia.org/wiki/Mut>
Mut - Wikipedia Mut From Wikipedia, the free encyclopedia Jump to navigation Jump to search For other uses, see Mut (disambiguation) . Not to be confused with Maat . Mut A contemporary image of goddess

[Amun - Wikipedia](#)
<https://en.wikipedia.org/wiki/Amun>
Amun - Wikipedia Amun From Wikipedia, the free encyclopedia Jump to navigation Jump to search For other uses, see Amun (disambiguation) . "Amen Ra" redirects here. For the

Metric clustering:

Query 1: Narmer Palette

Resulting expanded queries and search results

← → ↻ ⓘ localhost:8089/HistoricalArtifactsSearch/ ☆ 📄 👤 ⋮

Apps 📱 Set up Basic Auth... 📖 Understanding LST... 🔄 serverless-applicati... 🌐 awslabs/serverless-... 🔍 Find Training | AWS... 🌐 relation » 📁 Other bookmarks

narmer palette Search

Showing results for - *narmer,palette,fig,tombs,org,Connor,presence,catfish,*

[Narmer - Wikipedia](#)
<https://en.wikipedia.org/wiki/Narmer>
Narmer - Wikipedia Narmer From Wikipedia, the free encyclopedia Jump to navigation Jump to search Ancient Egyptian pharaoh of the Early Dynastic Period Narmer Menes Verso of Narmer Palette Pharaoh Rei

[Narmer Palette - Wikipedia](#)
https://en.wikipedia.org/wiki/Narmer_Palette
Narmer Palette - Wikipedia Narmer Palette From Wikipedia, the free encyclopedia Jump to navigation Jump to search Narmer Palette Both sides of the Narmer Palette Material siltstone Size c. 64 cm x 42

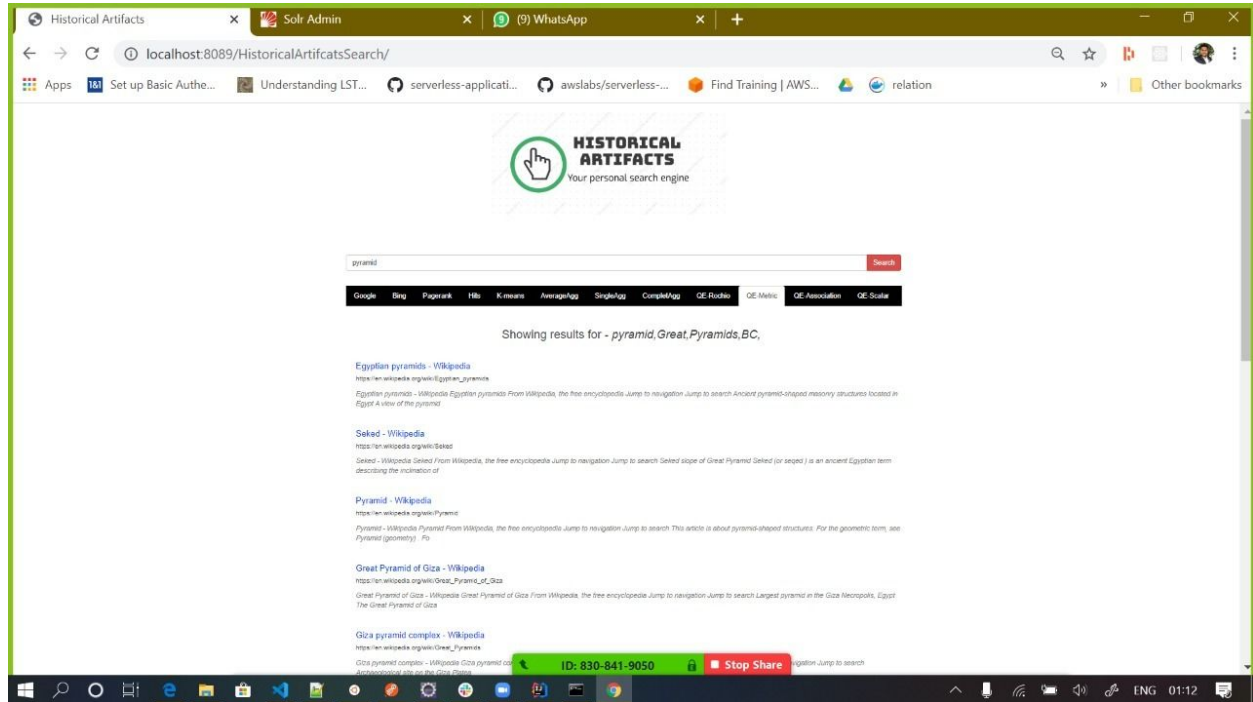
[Palette of King Narmer – Smarthistory](#)
<https://smarthistory.org/palette-of-king-narmer/>

ARIHANT CHHAJED

NETID: ARC180006

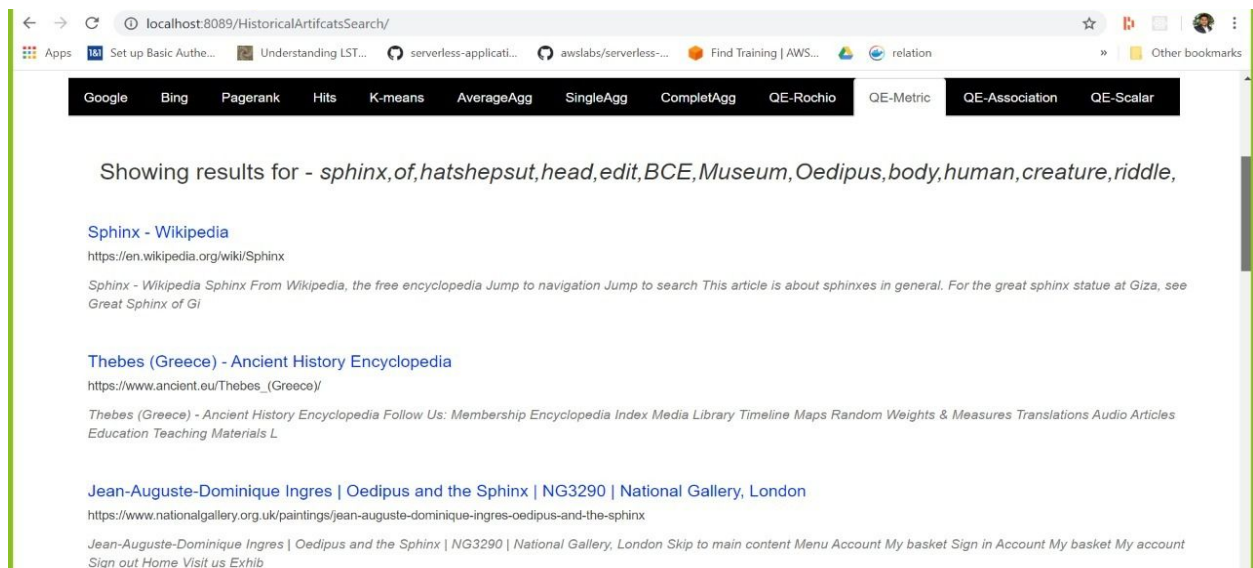
Query 2:pyramid

Resulting expanded queries and search results



Query 3:Sphinx of Hatshepsut

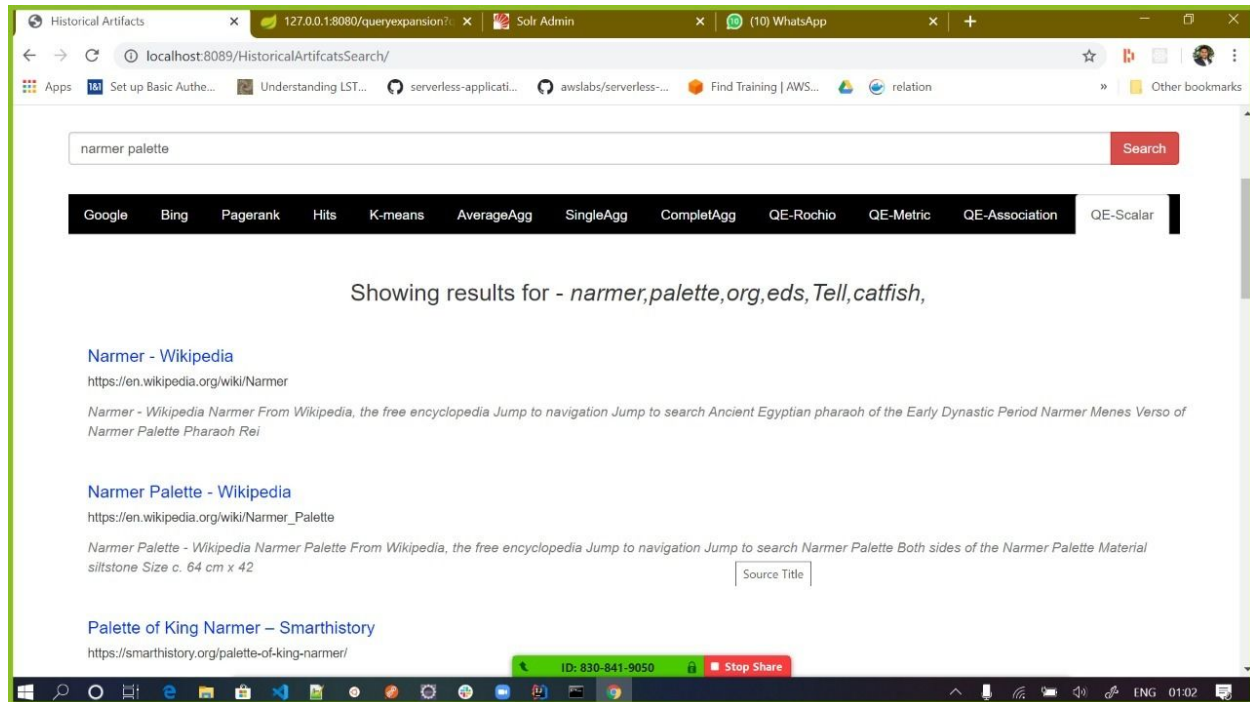
Resulting expanded queries and search results



Scalar clustering:

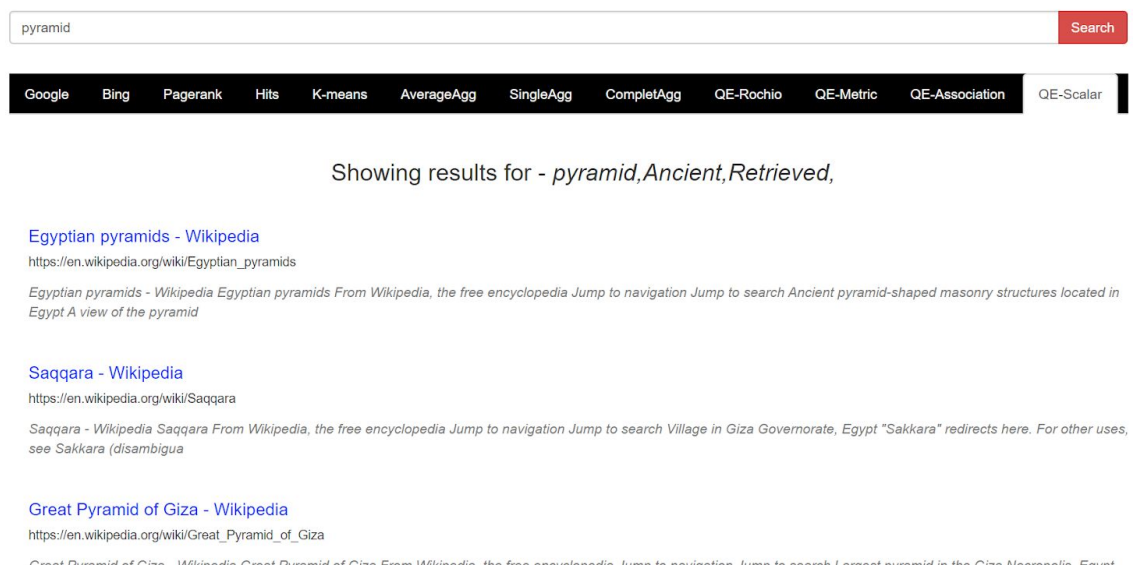
Query 1: Narmer Palette

Resulting expanded queries and search results



Query 2: pyramid

Resulting expanded queries and search results



Query 3: Sphinx of Hatshepsut

Resulting expanded queries and search results

[Google](#)
[Bing](#)
[Pagerank](#)
[Hits](#)
[K-means](#)
[AverageAgg](#)
[SingleAgg](#)
[CompleAgg](#)
[QE-Rochio](#)
[QE-Metric](#)
[QE-Association](#)
[QE-Scalar](#)

Showing results for - *Sphinx,of,Hatshepsut,cultural,Munich,Honarhay,city,atop,boat,*

[Sphinx - Wikipedia](#)
<https://en.wikipedia.org/wiki/Sphinx>
Sphinx - Wikipedia Sphinx From Wikipedia, the free encyclopedia Jump to navigation Jump to search This article is about sphinxes in general. For the great sphinx statue at Giza, see Great Sphinx of Gi

[Mut - Wikipedia](#)
<https://en.wikipedia.org/wiki/Mut>
Mut - Wikipedia Mut From Wikipedia, the free encyclopedia Jump to navigation Jump to search For other uses, see Mut (disambiguation) . Not to be confused with Maat . Mut A

COLLABORATION WITH UI PERSON (Krishan Duhan):

1. The query expansion algorithm is implemented and hosted in a separate server. The input to the Query expansion API is the initial query string made by the user and the type of query expansion algorithm to be used.
2. Final output from the API is the expanded query string. This expanded string is then used to retrieve a new set of results from the Solr API.
3. The information for the JAVA API call to this service along with the structure of the results returned is given to the UI person (Krishan Duhan) and these results are rendered on the UI as a json object.

COLLABORATION WITH INDEXING PERSON (Arpita Dutta):

1. Solr is used for Page Ranking and indexing. The server for pagerank and indexing exposes an API for accessing the indexed results. The method creates the URL for a select query in Solr, and fetches the output for that query in Solr.
2. Calling the Solr server directly using the information provided by the student responsible for indexing and searching (Arpita Dutta).
3. API also allows us to set some control parameters for e.g number of pages to be returned or format of the response. We have set the number of pages to be returned to be 15 and JSON format is used for the exchange of the data.

4. The python server API call returns results in json format for the PageRank and HITS algorithms. Information about this python API was collected from Arpita to make the API call. Since the PageRank and HITS algorithm is dependent on the indexing results from Solr, results from Solr API were sent to this API via POST and the response was collected back.
5. The query expansion results from the local server optimized along with documents retrieved from the Solr service and ranked according to the page-rank algorithm from the python server are all displayed on the UI.

BACKEND INTEGRATION (Arihant Chhajed)

1. Various modules like indexing , clustering, page ranking, query expansion are put together using Java Springboot REST api and python flask api.
2. Communication channel between these services was implemented using REST api based protocols.
3. All the data is transmitted into standard JSON format so that the frontend designer can use that for UI presentation.
4. Below are the technology stack used for various services in this service engine:-
 - a. **HITS/ PAGE Ranking** :- Python flask server is used to provide the page ranking and Hits data to the front end application since the program was written in python by my colleague Arpita Dutta.
 - b. **Clustering**:- Python flask server is used to provide the data in JSON format to the UI by using a clustering program provided by Pawan Vaidya.
 - c. **Query Expansion**:- Java Springboot was deployed on tomcat server for serving expanded query to the front end application.
 - d. **API Handler and Servlet Handler**:- API handler was written as a part of Java dynamic web application for handling various api calls to internal as well as external service like google, bing , solr, etc.

Service	Endpoint
HITS/PageRanking	POST: http://localhost:5000/getHits/<Boolean>
Clustering	POST: http://localhost:5001/clustering/<type>
Query Expansion	GET: http://localhost:8080/queryexpansion?q=<query>&qe=<type>
Dynamic Web Application	Service Endpoint: http://localhost:8089/HistoricalArtifactSearchEngine

Description of Individual Files

1. APICallHandler.java : Handles calls to individual APIs. HttpURLConnection class is used to make POST requests to individual APIs such as for the methods that are used - Bing call, Google Call, Solr call, Query Expansion Call and Hits call along with Clustering call.
2. ClusterServletHandler.java : Handles server requests that tie to K-means and agglomerative clustering at port 5001
3. QueryExpansionServletHandler.java : Handles server requests related to Query Expansion queries at port 8080
4. ServletHandler.java : Handles the various back-end calls to the JAVA servlets for Solr service (8983 port), Google, Bing, PageRank, HITS, Cluster, Query Expansion.

DISCUSSION:

The query expansion codes written initially were to be modified to be programmed as API endpoints and we had to collaborate so results can be rendered in the UI efficiently.

CONCLUSION:

Query Expansion is certainly a very useful technique to provide relevant page results. From the queries tested we were able to find that the clustering and query expansion results were much more relevant than the initial ones displayed. Hence it was helpful to perform these last two steps so as to enhance our search engine.

NIKITA VISPUTE
ARPITA DUTTA
KRISHAN KUMAR
PAVAN VAIDYA
ARIHANT CHHAJED

NETID:NXV170005
NETID:AXD170025
NETID:KXK180034
NETID:PAV180001
NETID:ARC180006

TEAM DISCUSSION:

1. Because of COVID-19, we were unable to meet in person with the whole team and hence had a brief discussion via conference call regarding what are the technology stacks we will be using. Once we started, we ran into initial issues with the crawling and indexing framework Apache Nutch and Solr installations which affected our initial schedule for completing tasks.
2. We started implementing the crawler with Scrapy and BeautifulSoup. But, the issue that we faced was we were not able to get dynamic content from the websites as most of the websites use JavaScript / Ajax to load the content. We decided to use open source crawler Apache-Nutch which was able to get dynamic content using a selenium plugin. Once the crawling started we discussed the type of output required from crawling so that indexing was read and processed on it.
3. While implementing Scrapy with python we faced the issue of how the output from crawling should be generated and what can be used for indexing, but once we started using Nutch and Solr open source, this got resolved.
4. After that we discussed how we will be setting up the UI and what languages would be easier to integrate the clustering, indexing and query expansion part with the UI. We had multiple discussions about how the expanded query and result should be displayed on the UI, also how to customize search variables/tabs, how to implement pagerank, HITS in combination with clustering and query expansion.
5. We did daily Zoom/Microsoft Teams meetings, sharing screens for quick communication and checking the workflow of the project as well as what needs to be worked on. For quick progress we also used whatsapp group to communicate.
6. The clustering and query expansion codes written initially were to be modified to be programmed as API endpoints and we had to collaborate to make the results can be rendered in the UI efficiently.

NIKITA VISPUTE
ARPITA DUTTA
KRISHAN KUMAR
PAVAN VAIDYA
ARIHANT CHHAJED

NETID:NXV170005
NETID:AXD170025
NETID:KXK180034
NETID:PAV180001
NETID:ARC180006

TEAM CONCLUSION:

1. Apache Nutch is one of most scalable tools and can be plugged with many different tools and used in distributed mode without having to make any major changes. But some more improvements can be done to improve the speed and efficiency by running it in a distributed mode or cloud environment.
2. Once the crawler worked using Apache Nutch Solr was used for indexing which was easily integratable with Nutch. The main part where time was consumed was writing a script to extract the inlink and outlink information and create a web graph from the same so as to implement the pagerank and HITS algorithms.
3. The initial UI was quite plain and simple, as we got to the enhancement part, we were able to add a few more modifications to it to make it look better and incorporate all the tabs for clustering and query expansion.
4. Clustering and Query Expansion is certainly a very useful technique to provide relevant page results. From the queries tested we were able to find that the clustering and query expansion results were much more relevant than the initial ones displayed. Hence it was helpful to perform these last two steps so as to enhance our search engine. It is also heavily dependent on the initial result set of documents, hence, some improvement can be done to reduce that dependency.