Project Name: ExpertSearch v2.0 (Auto crawler integration with faculty search improvements)

Description: The UIUC ExpertSearch system has several features such as faculty search, filtering criterions, search results (with options to open faculty bio pages, emailing, location info), pagination etc. As a team, we did a deep analysis of the ExpertSearch capabilities and found several deficiencies that need to be addressed to make it a better search system. The deficiencies include lack of accuracy, lack of relevant search results and inconsistencies in the search results. These deficiencies can be addressed using the text retrieval and text mining techniques that will improve the overall search experience in the ExpertSearch system. The team will be involved in implementing features such as converting unstructured dataset to structured dataset (e.g., csv, json), identifying the key topics (e.g., areas of interest) for each of the faculties and display in the search result, introducing an admin interface that would classify faculty pages and finally improving some of the existing features in the search page for better search experience.

Which tasks have been completed? Which tasks are pending? Overall Completion: **52%**

Epic	User Story	Progress
Crawling and Scraping Epic Completion – **73%**	Crawler Implementation for a given webpage url **Description**: In the admin interface when admin inputs an url, this story takes the url as input and scrapes the page and extracts the faculty biodata. We also implemented intelligent logic in scraper to find right faculty page if the base url has links that leads to multiple faculty related pages.	100%
	Adding admin interface for web page indexing **Description**: As our project scope doesn't include auto crawler features for the entire web, the admin interface we are implementing in ExpertSearch system is to allow the admin to enter base url of the universities and based on valid/invalid university email (different story) the admin interface will fetch the url to the crawler module to scrape faculty data.	100%
	Displaying accepted/rejected web page based on url **Description**: When admin enters the base url, this module will check if the url is a valid university url. If yes, the module forwards the url for crawling and scraping the faculty pages. If not, the module lets the admin know that base url doesn't belong to a university or no faculty page found.	20%
Search Experience Enhancement Epic Completion – **13%**	Build a structured dataset from Unstructured datasets **Description**: In current ExpertSearch system the faculty data are stored as unstructured data as a file. We are implementing a functionality that will convert the unstructured data to structured data. For e.g., using text mining and text retrieval techniques we are planning to extract fields like Faculty Name, Department, University, Area of Interests, email, phone, etc, and store them in a structured form (either in csv, or database etc.) This will enhance the overall search experience.	20%
	Enhance the search experience with relevant search results **Description**: Based on search input, we will look up all biodata from structured dataset and implement a ranking function using metapy. Based on ranking results we will extract corresponding fields from the structured data and display as search results. We will enhance filter based searching feature too where user can get better accuracy because of structured dataset.	10%
	Better consistency in displaying links such as email, phone etc. leveraging the structured data **Description**: The current expert search system doesn't show contact info (email, etc.) consistently across the search results even if the faulty page does have the data. Our improved scraping and structured data along with improved data display logic will increase the consistency in displaying the fields in search results.	10%

Topic Mining Epic Completion – **50%**	Using text mining techniques to extract the Areas of interest for a given faculty based **Description**: Using text mining methods we are planning to generate “Areas of Interests” data from the faculty bio. We are using guided LDA algorithm and Gensim/NLTK libraries to explore other topic mining features and we will be experimenting with parameters to generate relevant topics.	**80%**
	Display Areas of Interest in the faculty search result **Description**: We are enhancing the front end of ExpertSearch to display faculty search results along with additional relevant fields such as faculty areas of interest and few more.	**20%**
Deployment Epic Completion – **66%**	Understand and Install current ExpertSearch System **Description**: Install and explore the ExpertSearch system and understand the features and functionalities (both frontend and backend). Experiment with code changes etc.	**100%**
	Deploy code into Heroku **Description**: As ExpertSearch is web-based framework, we will do our deployments in Heroku and make it public. We will also do a git PR on the existing original ExpertSearch repo. But launching as an improved system and others to validate, we will separately host ExpertSearchv2.0 in Heroku.	**50%**
	Validation Exercises **Description**: As we do development and deployment, we are doing multiple rounds of verification and validation and some will require integrated end-to-end validation steps.	**50%**
Documentation Epic Completion – **60%**	Proposal Documentation **Description**: Project Proposal documentation.	**100%**
	Progress Report Documentation **Description**: Project Progress documentation.	**100%**
	Final Project Report Documentation **Description**: Detailed Final Project Report documentation.	**40%**
	Demo Video Presentation and Editing **Description**: Video presentation demonstrating the overall project, designs and tech approaches, features and functionalities, learnings and improvement areas etc.	**Not Started**

Progress colors: 🟩 → Completed, 🟨 → 50% or more completed, ⬜ → Less than 50% completed

Are you facing any challenges?

- When trying to install ExpertSearch in Windows environment, within the ExpertSearch system Ranker.Score function wasn’t working and it seems it is owing to missing query and index files in the config.toml. This seems to be windows specific execution issue. We are trying with exploring python virtual environments and other possible fixes to see if we can launch this system in Windows system too. However, not being able to do that is not a complete clocker for us. We were able to use the WSL feature in Windows and able to proceed with the ExpertSearch installation and execution.

- When trying to install ExpertSearch in macos environment, we initially tried on install in python2.7 conda virtual environment. When we tried to launch the system using the command "`gunicorn server:app -b localhost:8095`", we encountered a RankingFunction metapy related reference issue, which is shown below

  ```
  AttributeError: module 'metapy.metapy.index' has no attribute 'RankingFunction'
  ```

  We also decided to give python3.5 a try.

- When trying to install ExpertSearch in macos environment, python3.5 conda virtual environment and tried to launch the system using the command "`gunicorn server:app -b localhost:8095`", we encountered a Json decoding issue, which is shown below:

  ```
    File "/Users/usaha/mcs/08_text_information_systems/Project/repo/ExpertSearch/server.py", line 62, in search
      data = json.loads(request.data)
    File "/Users/usaha/opt/anaconda3/envs/python3.5/lib/python3.5/json/__init__.py", line 312, in loads
      s.__class__.__name__))
  TypeError: the JSON object must be str, not 'bytes'
  ```

  But as we decided to run the project on python3.5 we ran after the issue for resolution and we were able to fix the issue by adding by decoding the request data in the `json.loads` function call in `server.py`. Below is the change we did:

  Before change in `server.py`:

  ```
  data = json.loads(request.data)
  ```

  After change in `server.py`:

  ```
  data = json.loads(request.data.decode("utf-8"))
  ```

  With the above change, we were successfully able to install and run the ExpertSearch system.

- We are facing challenges while trying to build a generic web crawler to scrape Faculty bio pages from a given University Home page url. This is because different websites maintain different HTML syntaxes and tags. We are still working on this.

- For extracting areas of interest data from the faculty bio data, we were doing topic mining on the data. We used NLTK topic mining algorithms. For a given dataset we noticed from all the topics that library extracted few of them were not relevant but was still getting a high score. We reached out the Professor/TAs for guidance and TA responded us by trying the guided LDA topic mining approach. So, we are now trying a 2-phase implementation. In phase 1 we are extracting all the topics for a faculty bio data and then on phase 2 we are using the list of topics (that we received from phase 1) as input and calling guided LDA to get more relevant topics. We are still exploring with some parameters, but we at least got a direction.