Luke Staib: U39863533 3-27-21
EC500 Professor Osama

Homework 2: Final Operations for the Secure File Uploader/Ingest, Text NLP Analysis, and Newsfeed Ingest

Secure File Uploader/Ingest

- API Type: Entity-based
- Operations:
    - UploadFiles(userID, file_in, fid, authors, creation_time)
        - Will upload file to account determined by user's ID (userID), method of upload through Flask
        - "file_in" is a file object obtained from the web interface
        - "fid" is the new file's ID
        - "authors" and "creation_time" are the variables entered by the user on the file upload screen. "authors" is the value that holds the authors of the file and "creation_time" holds the time of creation of the original article
        - Will fail if userID and fid are not integers, and authors and creation_time are not strings
        - Will fail if userID does not correspond to a user in the database
        - Will fail if file cannot be created
        - On success, the new file is returned to the frontend to update the database
    - FileDelete(fileID)
        - Will delete file from database with matching file ID (F_ID)
        - Will fail if fileID is not an integer
        - Will fail if "delete_one" command fails (MongoDB deletion command)
    - FileEditName(fileID, authors, creation_time)

        - Will allow user to change the authors and creation_time that they entered when creating the file
        - Will fail if fileID is not an integer, or if authors or creation_time are not strings
        - Will fail if “update_one” command fails (MongoDB edit command)
    - DiagnosticsUploader()
        - Tests CPU usage and network traffic/bandwidth usage
        - Can be used to test memory usage if the “cProfile.run()” command is put into the frontend
- Data:
    - Text data: TXT, PDF, DOCX
    - Can handle large files, only these three filetypes
    - Only articles
- Status:
    - Before upload
        - Web application gives user option to upload a specific file, along with entering authors and the creation date
    - During upload
        - Web application loads while uploading, showing that the webpage is loading at the bottom of the screen
    - After successful upload
        - User is sent back to their homepage and a message is shown that tells the user that their file was successfully uploaded
    - Error during upload process
        - User is sent back to their homepage and a message is shown that tells the user that an error occurred
        - Error information in development logs
    - Files uploaded display
        - Will display files that have been uploaded, currently on user’s account
        - Will be able to delete, edit entered aspects of files

        - A user can see the name, upload date, creation date, and NLP analysis and status of that file

Text NLP Analysis

- API Type: Procedure-based
- Operations:
    - ConvertFilesToText(userID, file_in, filetype)
        - Convert file data into TXT data to be used in text analysis -> Google NLP API commands
        - Will fail if userID is not an integer or filetype is not a string
        - Segment for testing “test file txt”, “test file pdf”, and “test file docx” to test conversion with files I have locally
        - Converts text in file and returns the text
    - CreateKeywords(text)
        - Takes in input from user and analyzes it using Google NLP’s analyze_entities() function. From the output of Google’s function, I take the keywords received and return them.
    - ObtainCategories(text)
        - Google NLP’s classify_text() function is used to generate categories based on the text entered. If there are no categories, return “N/A” to format in the web application.
    - AssessData(article_data, keywords, text)
        - This is where the program will create a sentiment of the text data entered using Google NLP’s analyze_sentiment() function. The sentiment results are returned including the text analyzed, and the score and magnitude of the text
    - DiagnosticsNLP()
        - Tests CPU usage and network traffic/bandwidth usage
        - Can be used to test memory usage if the “cProfile.run()” command is put into the frontend
- Data:

    - Text data turned into TXT: TXT, PDF, DOCX
    - Input is converted into TXT to be analyzed by the system, outputted as keywords, categories, and sentiment
- Status:
    - In the web app, the user can select one of their uploaded files for analysis
    - Upon completion, the user will be sent to the homepage and a message will display telling the user whether the operation was successful or whether there was an error

Newsfeed Ingest

- API Type: Entity-based
- Operations:
    - DiscoverContent(search_text)
        - Takes in search query that user enters to search for relevant information on that topic using the New York Times Article Search API
        - Returns with up to 30 relevant searches if searches are found, otherwise nothing is displayed
    - DiagnosticsNewsfeed()
        - Tests CPU usage and network traffic/bandwidth usage
        - Can be used to test memory usage if the “cProfile.run()” command is put into the frontend
- Data:
    - API takes in text based data, entered in a text field in the web application
    - API returns a list of articles that are parsed to display to the user
- Status:
    - API will display relevant articles (if articles are found) or nothing (if no articles are found) upon searching

Diagnostic and Logging Functions

- Covered individually in each module, they have separate diagnostic functions

- The Python logging library is used to store a log file for each module and display usage information, warnings, and errors