**PII INDENTIFIER**

**For Entity Recognition – Document Scarping:**

1. **For PDF Extraction(extract\_text\_from\_pdf(pdf\_path)):**

* *Description:* This function extracts text content from a PDF file.
* *Methodology:*

1. It utilizes the `fitz` module, presumably from PyMuPDF, to open the PDF file.

2. It iterates through each page of the PDF and extracts text using the `get\_text()` method.

3. The extracted text from each page is concatenated to the variable `text`.

4. If successful, it returns the concatenated text; otherwise, it raises an exception with an error message.

1. **For Text file Extraction(extract\_text\_from\_text\_file(text\_file\_path)):**

* *Description:* This function extracts text content from a Word document.
* *Methodology:*

1. It opens the text file in read mode with UTF-8 encoding.

2. It reads the entire content of the text file into the variable `text`.

3. If successful, it returns the extracted text; otherwise, it catches any exception, prints an error message, and returns an empty string.

1. **For Document Extraction(extract\_text\_from\_document(docx\_file\_path)):**

* *Description:* This function extracts text content from a Word document.
* *Methodology:*

1. It uses the `Document` class, presumably from the `python-docx` library, to open the Word document.

2. It iterates through each paragraph in the document and appends the text to the variable `text`.

3. If successful, it returns the concatenated text; otherwise, it catches any exception, prints an error message, and returns an empty string.

**For Entity Recognition:**

**1. CITIZENSHIP:**

* *Library:* Presidio Analyzer
* *Methodology:* Utilizes Presidio Analyzer with a custom pattern recognizer for citizenship entities, checking against a deny list of citizenship and immigration terms.

**2. PERSON:**

* *Library:* Presidio Analyzer
* *Methodology:* Applies Presidio Analyzer with a custom pattern recognizer for person names, incorporating a deny list of female names.

**3. CRIMINAL HISTORY:**

* *Library:* Presidio Analyzer
* *Methodology:* Leverages Presidio Analyzer with a custom pattern recognizer designed for recognizing criminal history entities, checking against a deny list.

**4. EMAIL:**

* *Library:* Regular Expressions (regex)
* *Methodology:* Employs a regular expression pattern to identify email addresses in the text.

**5. RELIGIOUS AFFILIATION:**

* *Library:* Presidio Analyzer
* *Methodology:* Uses Presidio Analyzer with a custom pattern recognizer for religious affiliations.

**6. SEXUAL ORIENTATION:**

* *Library:* Presidio Analyzer
* *Methodology:* Applies Presidio Analyzer with a custom pattern recognizer for sexual orientation entities, using a deny list of identity terms.

**7. DRIVER LICENSE NUMBER:**

* *Library:* Presidio Analyzer
* *Methodology:* Defines a specific pattern for driver's license recognition, employs Presidio Analyzer with a custom pattern recognizer.

**8. FINANCIAL ACCOUNT NUMBER:**

* *Library:* Regular Expressions (regex)
* *Methodology:* Defines a specific pattern for financial account recognition, employs Presidio Analyzer with a custom pattern recognizer.

**9. PHONE NUMBER:**

* *Library:* Regular Expressions (regex)
* *Methodology:* Defines a specific pattern for phone number recognition, employs Presidio Analyzer with a custom pattern recognizer.

**10. BIOMETRIC IDENTIFIER:**

* *Library:* Presidio Analyzer
* *Methodology:* Defines a specific pattern for biometric identifier recognition, employs Presidio Analyzer with a custom pattern recognizer.

**11. PATIENT ID NUMBER:**

* *Library:* Presidio Analyzer
* *Methodology:* Defines a specific pattern for patient ID recognition, employs Presidio Analyzer with a custom pattern recognizer.

**12. DATE OF BIRTH:**

* *Library:* Presidio Analyzer
* *Methodology:* Defines a specific pattern for date of birth recognition, employs Presidio Analyzer with a custom pattern recognizer.

**13. MEDICAL HISTORY:**

* *Library:* Presidio Analyzer
* *Methodology:* Uses Presidio Analyzer with a custom pattern recognizer for medical history entities, checking against a deny list.

**14. GENDER:**

* *Library:* Presidio Analyzer
* *Methodology:* Applies Presidio Analyzer with a custom pattern recognizer for gender entities, using a deny list.

**15. TITLE:**

* *Library:* Presidio Analyzer
* *Methodology:* Applies Presidio Analyzer with a custom pattern recognizer for title entities, using a deny list.

**16. TAX ID:**

* *Library:* Regular Expressions (regex)
* *Methodology:* Defines a specific pattern for tax ID recognition, employs Presidio Analyzer with a custom pattern recognizer.

**17. LOCATION:**

* *Library:* Presidio Analyzer (with custom regular expression)
* *Methodology:* Defines a specific pattern for location recognition, employs Presidio Analyzer with a custom pattern recognizer.

**18. DATE:**

* *Library:* Datefinder
* *Methodology:* Utilizes datefinder to extract dates from the text.

**19. AAA NUMBER:**

* *Library:* Custom Pattern Recognition
* *Methodology:* Defines a specific pattern for AAA number recognition, employs Presidio Analyzer with a custom pattern recognizer.

**20. ORGANIZATION:**

* *Library:* Spacy
* *Methodology:* Uses SpaCy to recognize organizations in the text.

**Flask Input-Output Details:**

**Input:**

The Flask application defines a route `/extract\_entities` that accepts HTTP POST requests. The expected input can be provided either as JSON in the request body or as form data. The required parameters are as follows:

* *`input\_file\_path`:* Path to the file containing the text data to be analyzed. (Optional, either this or `input\_text` must be provided.)
* *`input\_text`:* The actual text data to be analyzed. (Optional, either this or `input\_file\_path` must be provided.)
* *`entities\_to\_extract`:* A list of strings specifying the types of entities to be extracted from the input data. The available entity types include 'PERSON\_NAME', 'TITLES', 'DATES', 'LOCATION', 'EMAIL', 'ORGANIZATION', 'CITIZENSHIP', 'PHONE\_NUMBER', 'CRIMINAL\_HISTORY', 'RELIGIOUS\_AFFLICATION', 'MEDICAL\_HISTORY', 'SEXUAL\_ORIENTATION', 'TAX\_ID', 'DRIVER\_LICENSE', 'BIOMETERIC\_IDENTIFIER', 'PATIENT\_ID', 'GENDERS', 'AAA\_NUMBERS'.

**Usage:**

* + Send a POST request to `http://127.0.0.1:5000/extract\_entities'.
  + Include either `input\_file\_path` (with the file path) or `input\_text` (with the actual text) in the request data.
  + Optionally include `entities\_to\_extract` to specify the types of entities to extract.
  + Receive the extracted entities in the JSON response.

***Additional Note:***

Logging has been used for improved traceability and debugging