**REPORT**

**Problem Statement:**

**Business Contract Validation –**To Classify Content within the Contract Clauses and Determine Deviations from Templates and highlight them.

**Introduction:**

When validating a business contract by classifying content within clauses and identifying deviations from templates, the process involves meticulous scrutiny. Each clause is categorized according to its purpose, such as defining obligations, specifying terms, or outlining responsibilities. By comparing these clauses against established templates or standard industry practices, any discrepancies or variations can be pinpointed and highlighted. This method ensures that the contract is not only comprehensive but also aligns closely with best practices, enhancing its clarity, enforceability, and legal compliance. Through this systematic approach, potential risks and ambiguities are minimized, facilitating smoother negotiations and clearer expectations between parties involved.

**Unique Idea Brief:**

The application is designed to assist in validating business contracts by classifying the content within the contract clauses, determining deviations from predefined templates, and highlighting these deviations. This unique idea leverages natural language processing (NLP) and text comparison algorithms to automate the contract review process, ensuring consistency and accuracy while reducing the time and effort required for manual reviews.

**Team members and contribution:**

1.VELURI BHANU TEJA SAHU: Flask App Setup and File Handling

Tasks:

🡪Set up the Flask application.

🡪Implement the file upload functionality.

🡪Save uploaded files to the appropriate directory.

🡪Extract text from uploaded PDF and image files.

2.A.SARAYU REDDY: Text Extraction and Processing

Tasks:

🡪Implement text extraction from PDF using PyMuPDF.

🡪Ensure that the extracted text is in a clean and readable format.

3. Madineni Keerthi Varshini: Comparison with Template

Tasks:

🡪Implement the function to compare extracted text with the contract template.

🡪Highlight deviations using CSS classes (highlight-red and highlight-green).

🡪Extract and format important contents to display deviations.

4. K Preethi: Highlighted Contract Terms

Tasks:

🡪Implement the function to highlight specific terms in the contract text.

🡪Ensure that "party," "agreement," "confidentiality," "Party A," and "Party B" are highlighted in the text.

🡪Format the highlighted contract for display.

5.PATAN SYED SAB VALI: Entities Extraction and Results Display

Tasks:

🡪Implement the function to extract entities using spacy.

🡪Format the extracted entities for display in a table.Create the HTML templates (upload.html and results.html) to display results.

🡪Ensure proper CSS styling for highlighting and tables.

Features Offered:

* PDF File Upload:

Allows users to upload PDF files of business contracts.

* Text Extraction:

Extracts text from the uploaded PDF files using PyMuPDF.

* Template Comparison:

Compares the extracted text against a predefined contract template to

identify deviations.

* Clause Highlighting:

Highlights important clauses and terms within the contract text.

* Deviation Highlighting:

Highlights deviations from the template in different colors for easy

identification (e.g., red for missing clauses, green for added clauses).

* Named Entity Recognition (NER):

Identifies and extracts key entities (e.g., names, dates, addresses) from

the contract text using spaCy.

* Web Interface:

Provides a user-friendly web interface for uploading files and displaying

results.

**Process flow:**

* User Uploads Contract:

The user navigates to the web application and uploads a PDF file of the business contract.

* Text Extraction:

The uploaded PDF is processed to extract its text content using the extract\_text\_from\_pdf function, which utilizes PyMuPDF.

* Template Comparison:

The extracted text is compared with a predefined template using the extract\_important\_contents function. The differences are highlighted to show deviations.

* Highlighting Terms:

Specific important terms and clauses within the contract text are highlighted using the highlight\_contract\_terms function.

* Entity Extraction:

The text is processed with spaCy to extract named entities (such as names, addresses, dates) using the extract\_entities function.

* Results Displayed:

The processed text, highlighted terms, deviations, and extracted entities are displayed on the results page.

* Flask:

A lightweight WSGI web application framework in Python used for building the web interface.

* PyMuPDF (fitz):

A library used for reading and extracting text from PDF files.

* spaCy:

An advanced NLP library in Python used for named entity recognition.

* difflib:

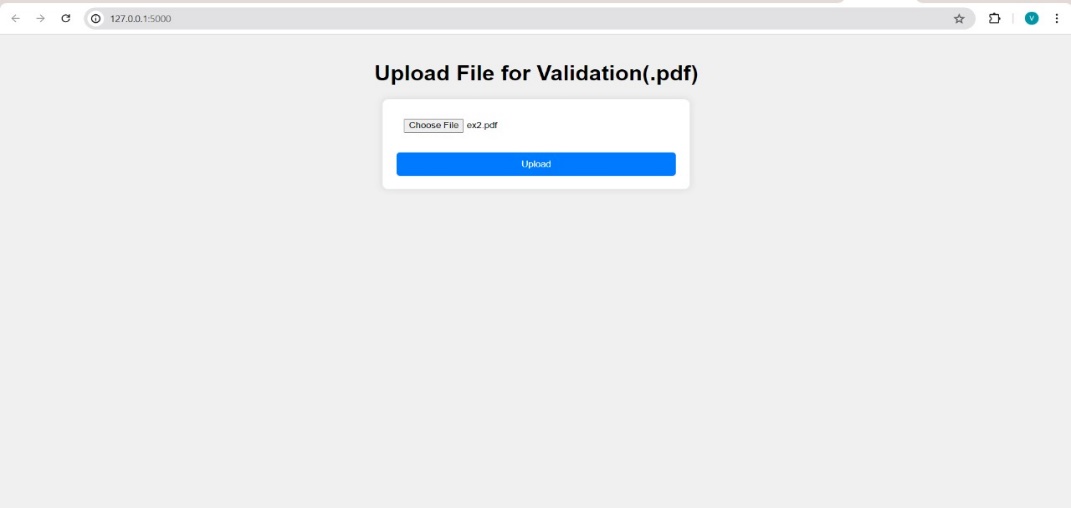
A module for comparing sequences, used to compare contract text with the template and highlight differences.

* HTML/CSS:

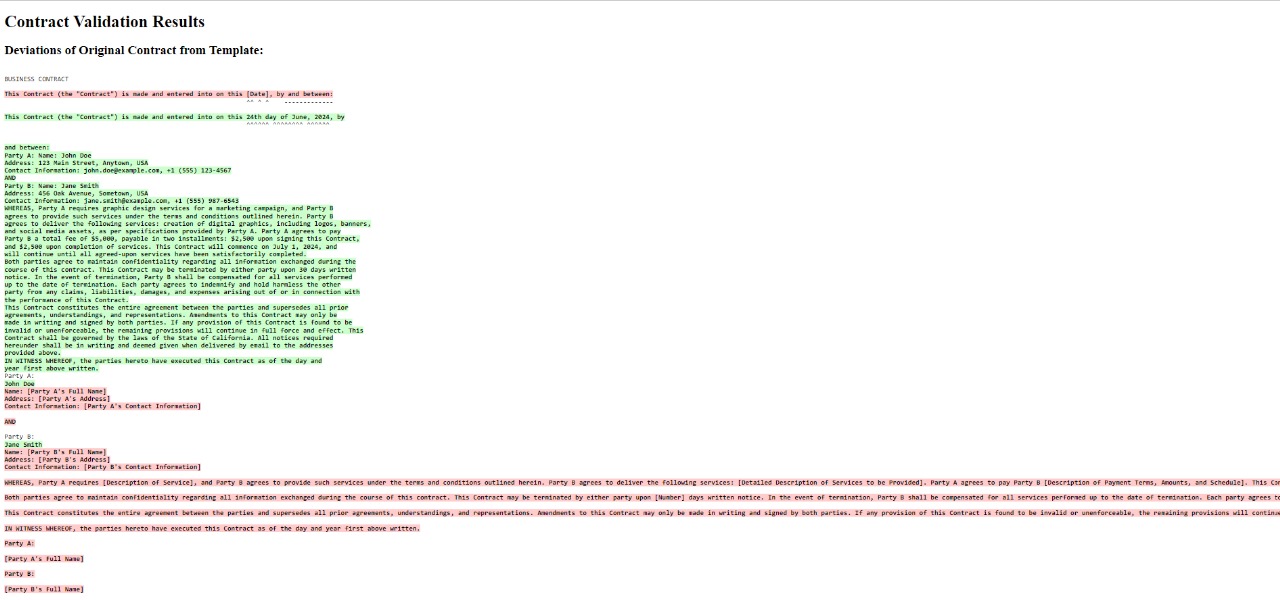
Used for creating the web interface and styling the highlighted text and results.

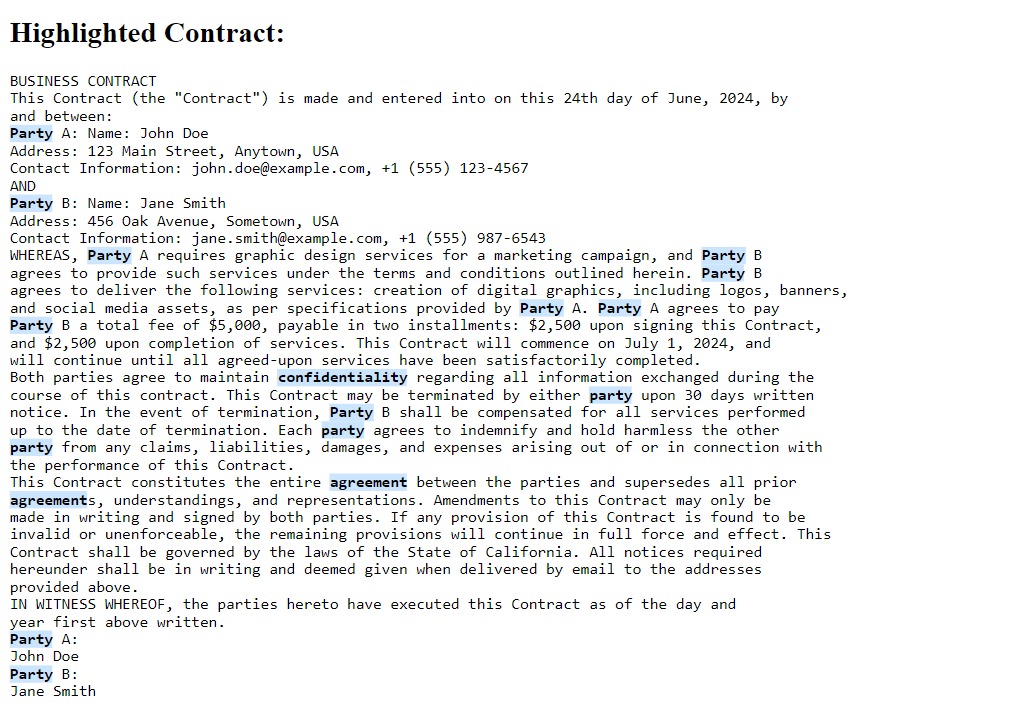
The combination of these technologies allows for a robust and efficient system to automate the contract validation process, enhancing accuracy and saving time.

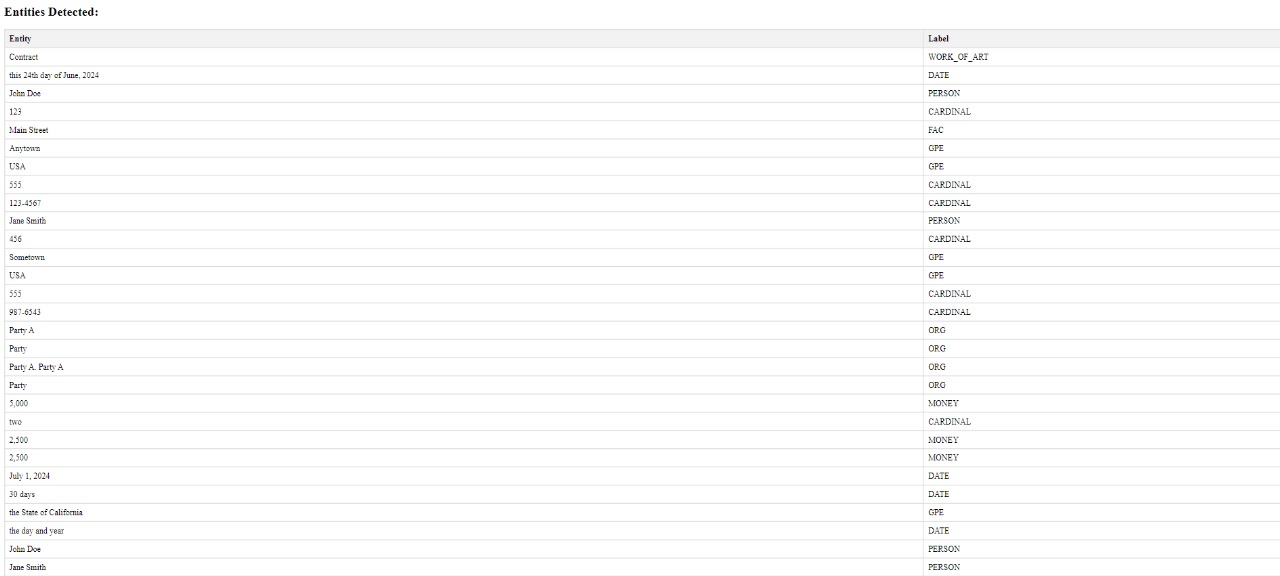
Conclusion:



SAMPLE OUTPUT:







* This project demonstrates the potential of combining various technologies to solve real-world problems effectively. By automating the tedious and error-prone process of contract validation, the application not only saves time and resources but also ensures higher accuracy and reliability. The collaborative effort of the team members, each bringing their expertise to the table, has resulted in a robust and user-friendly application that meets the needs of businesses dealing with complex contracts.
* The successful implementation of this project showcases the importance of integrating machine learning and natural language processing in business applications, paving the way for more innovative solutions in the future.