Business contract validation

Ajith kumar.S

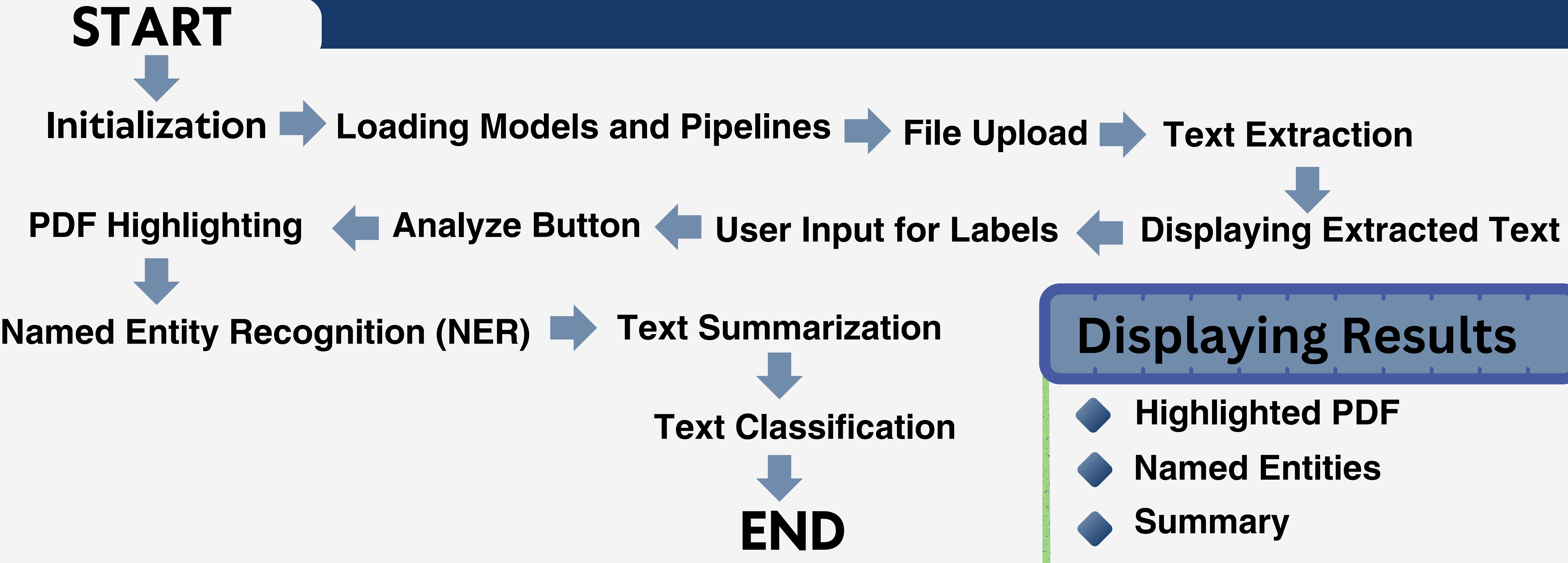
Unique Idea Brief (Solution)

- Lazy Loading with Caching
- Combining NER with User-Defined Labels
- Highlighting PDF with Custom Annotations
- Text Extraction from Multiple Formats

Features Offered

1	File Upload Support	5	Highlighting in PDF
2	Text Extraction	6	Summarization
3	Named Entity Recognition (NER)	7	Zero-Shot Text Classification
4	User-Defined Labels	8	Session State Management

Process flow



START

Initialization

Loading Models and Pipelines

File Upload

Text Extraction

Displaying Extracted Text

User Input for Labels

Analyze Button

PDF Highlighting

Named Entity Recognition (NER)

Text Summarization

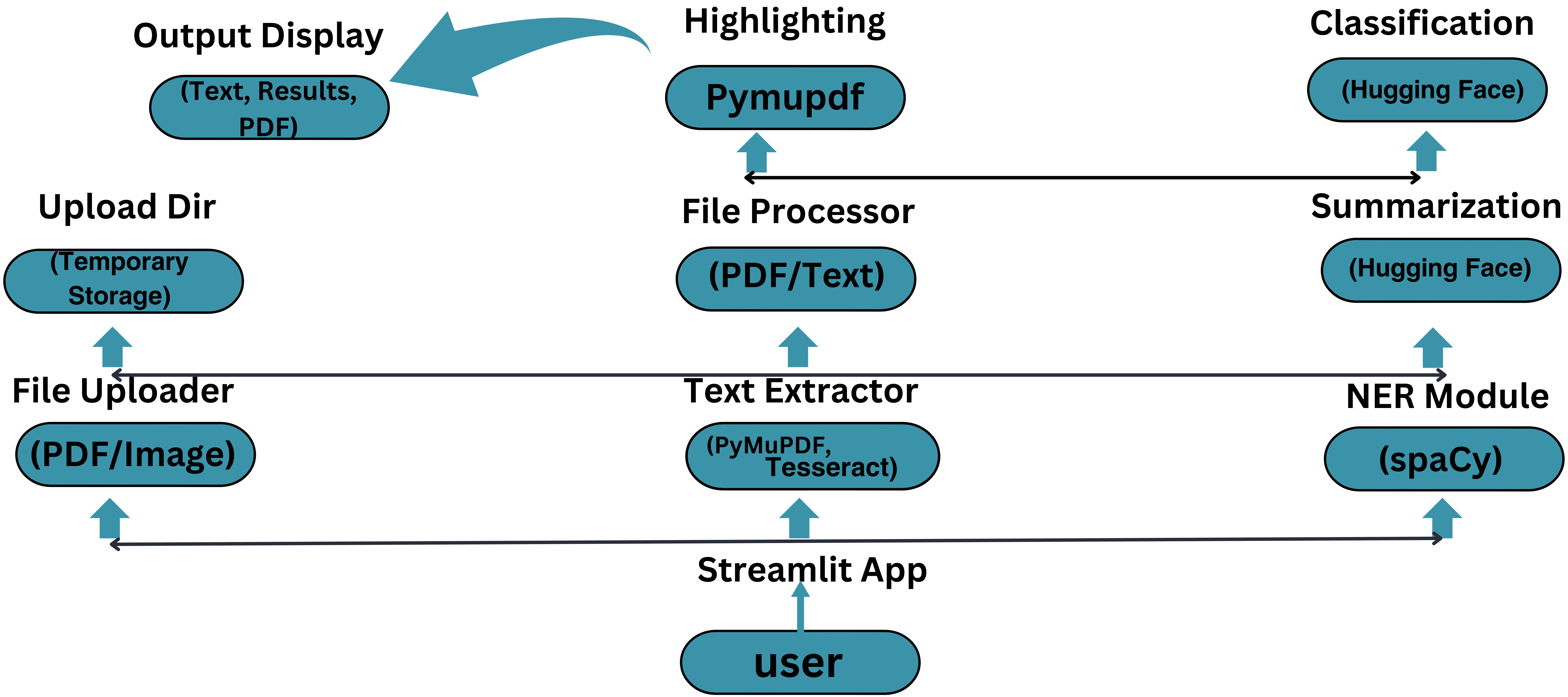
Text Classification

END

Displaying Results

- Highlighted PDF
- Named Entities
- Summary
- Classification Results

Architecture Diagram



Output Display

(Text, Results, PDF)

Highlighting

Pymupdf

Classification

(Hugging Face)

Upload Dir

(Temporary Storage)

File Processor

(PDF/Text)

Summarization

(Hugging Face)

File Uploader

(PDF/Image)

Text Extractor

(PyMuPDF, Tesseract)

NER Module

(spaCy)

Streamlit App

user

Technologies used

- ★ Web Framework
- ★ PDF Processing
- ★ OCR Model
- ★ NLP Models
- ★ Data Encoding
- ★ Text Processing
- ★ Filesystem Handling

Conclusion

★ Text Extraction:

Handles both PDF and image files.

Extracts text using PyMuPDF for PDFs and Pytesseract for images.

★ Text Analysis:

Identifies named entities using Spacy.

Summarizes the text with the Hugging Face Transformers library.

Classifies text based on user-selected labels using zero-shot classification.

★ User Interface:

Streamlit provides an easy-to-use web interface.

Users can upload files, select predefined or custom labels, and view the results.

★ Highlighted Results:

Highlights entities and user-defined labels in the PDF.

Displays extracted text, summaries, classifications, and the highlighted PDF on the web page.

★ Efficiency:

Uses caching to improve performance and avoid redundant computations.