Comprehensive Prompt for Complex Document Translation & Validation

Context

This prompt is designed for enterprise environments (e.g., banks) to handle complex legal PDF documents that need to be translated while retaining original layout, structure, and compliance. The workflow involves: 1. Receiving a **source PDF** (legal document in source language). 2. Using **Gemini Pro** (or equivalent LLM) to translate the content, producing an **intermediate HTML** output. 3. Converting the translated HTML back into a **target PDF**, preserving tables, images, formatting, and references. 4. Performing **automated validation** including section alignment, COMET/METEOR scores, and missing section detection. 5. Ensuring only **enterprise-safe libraries** (Java, Python standard libraries, Apache PDFBox, iText, pdfplumber, pdfminer.six, BeautifulSoup, lxml) are used for parsing, validation, and rendering tasks.

Detailed Prompt

You are an AI assistant working inside a secure enterprise environment. Your task is to process a **source PDF** and a **target PDF** (translated version). Ensure the following: - Extract all text, tables, images, headers, and footers from both PDFs. - Align sections (title, clauses, annexures, signatures, etc.) between source and translated versions. - Highlight missing or extra sections in the translated file. - Score each section using **COMET** and **METEOR** for translation quality. - Ensure layout fidelity: formatting, indentation, bulleting, numbering, and page references must be preserved. - Output: 1. JSON report of alignment, quality scores, and discrepancies. 2. Annotated translated PDF (with highlights for missing/mismatched content). 3. Final compliance summary for audit purposes.

Suggested Code Structure

```
project-root/
■
■■■ src/
■   ■■■ extract/
■   ■   ■■■ pdf_extractor.py        # Extracts structured data (text, tables, images) from PDFs
■   ■   ■■■ html_generator.py       # Converts extracted data into clean HTML
■   ■
■   ■■■ translate/
■   ■   ■■■ gemini_integration.py   # Calls Gemini Pro with HTML for translation
■   ■   ■■■ alignment_checker.py    # Aligns sections and checks for missing/mismatched parts
■   ■
■   ■■■ validate/
■   ■   ■■■ scoring.py              # COMET & METEOR scoring per section
■   ■   ■■■ compliance_checker.py   # Ensures legal/compliance terms preserved
■   ■
■   ■■■ render/
■   ■   ■■■ pdf_renderer.py         # Converts translated HTML back to PDF
■   ■   ■■■ annotator.py            # Highlights discrepancies in target PDF
■   ■
■   ■■■ reports/
■   ■   ■■■ report_generator.py     # Generates JSON + PDF compliance reports
■   ■
■   ■■■ main.py                    # Orchestration entry point
■
■■■ tests/
■   ■■■ test_pdf_extractor.py
■   ■■■ test_alignment_checker.py
■   ■■■ test_scoring.py
■
■■■ configs/
■   ■■■ settings.yaml              # Config for paths, model parameters, thresholds
```

```
■
■■■ requirements.txt                # Dependencies (enterprise-safe only)
```

Enterprise-Safe Libraries

- **Python**: pdfminer.six, pdfplumber, BeautifulSoup, lxml, reportlab, difflib, re, json - **Java (optional)**: Apache PDFBox, iText - **Scoring**: COMET, METEOR (can be run via offline models or wrapped services) - **No external SaaS or non-audited open-source packages** allowed.