ClauseWise – Legal Document Analyzer

**Project Title**:  
ClauseWise – Legal Document Analyzer using IBM Model

**Team Name:**  
Team Interstellar

**Team Members:**

* Santhi Priya Merugu
* Sushma Varshitha Palivela
* Sowmya Mothika

# Phase-1: Brainstorming & Ideation

**Objective:**  
Build an AI-powered legal assistant to simplify and extract clauses from legal documents.

**Problem Statement:**  
Legal documents are complex and difficult for non-experts to understand. Extracting useful information manually is time-consuming.

**Proposed Solution:**  
A Streamlit + FastAPI based app that uses Hugging Face models(IBM Model) to extract, simplify, classify legal clauses and also generate results via google search and the source links to the relevant documents.

**Target Users:**  
Legal professionals, startups, HR departments, individuals dealing with contracts,policies and legal documents.

**Expected Outcome:**  
A working prototype capable of document upload, clause simplification, entity recognition, classification and direct legal question answering with the source links.

# Phase-2: Requirement Analysis

**Objective:**  
Define technical and functional requirements for ClauseWise.

**Technical Requirements:**  
- Python 3.9+  
- Backend: FastAPI  
- Frontend: Streamlit  
- Hugging Face Transformers API  
- No database initially

**Functional Requirements:**  
- Upload PDF/DOCX/TXT  
- Extract clauses  
- Simplify clauses using GPT2  
- Perform NER  
- Classify document type

**Constraints & Challenges:**  
- Managing long input tokens for models  
- Maintaining semantic accuracy in simplification  
- Responsive UI design in Streamlit

# Phase-3: Project Design

**System Architecture:**  
- Document Upload → Clause Extraction → AI Processing (Simplify, NER, Classify, Source Linking) → Display via Streamlit UI

- User Asks Legal Question → Automated Google Search → Relevant Source Link Generation → Summarized Answer Display → Streamlit UI Output

**User Flow:**

**For Document Analysis:**

Upload a legal document (PDF, DOCX, TXT)

Choose function: *Simplify*, *NER*, or *Classify*

View structured results: original + simplified clauses, recognized entities, or document type

Access suggested Google search links and references for deeper understanding

**For Legal Q&A:**

Enter a legal question in the input box

System performs automated Google search

Summarized answer with clickable source links is displayed

Option to explore full articles from trusted legal source.

**UI/UX Considerations:**

- Clause summary

- Highlighted named entities with color tags and tooltips

- Responsive layout for mobile and desktop users

- Tabs or toggles to switch between *Document Analysis* and *Legal Q&A*

- Embedded buttons for source links and Google searches

- Minimalist design with clear headings and functional sections

# Phase-4: Project Planning (Agile Methodologies)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint | Task | Priority | Duration | Deadline | Assigned To | Expected Outcome |
| Sprint 1 | Setup Environment & Models | 🔴 High | 6 hrs | Day 1 | Sowmya | HF Models integrated |
| Sprint 1 | Streamlit UI Skeleton | 🟡 Medium | 3 hrs | Day 1 | Sushma | Basic UI  ready |
| Sprint 2 | Clause Simplification | 🔴 High | 4 hrs | Day 1 | Santhi | Simplified outputs |
| Sprint 2 | NER Module | 🔴 High | 3 hrs | Day 2 | Santhi | Entities displayed |
| Sprint 3 | Testing & Bug Fixes | 🟡 Medium | 2 hrs | Day 2 | Team | Stable version |

# Phase-5: Project Development

**Technology Stack Used:**  
- Frontend: Streamlit  
- Backend: FastAPI  
- AI: Hugging Face Transformers  
- Libraries: pdfplumber, python-docx

**Development Process:**- Model setup and inference logic  
- Clause extraction pipeline  
- API integration and route creation  
- UI component design and testing

**Challenges & Fixes:**  
- Challenge: Model token limits → Fix: Split long texts  
- Challenge: Streamlit responsiveness → Fix: CSS layout tuning

# Phase-6: Functional & Performance Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Category | Test Scenario | Expected Outcome | Status | Tester |
| TC-001 | Functional | Simplify sample lease clause | Readable simplified clause | ✅ Passed | Sowmya |
| TC-002 | Functional | Run NER on NDA | Entities like 'Parties', 'Dates' | ✅ Passed | Sushma |
| TC-003 | Performance | Test response under 1s | Should return in <1s | ✅ Passed | Santhi |

# Conclusion:

ClauseWise enhances legal document analysis by not only simplifying clauses and extracting key entities, but also generating relevant source links and enabling automatic Google search for further reading. This helps users quickly verify legal terms, find case references, and deepen their understanding—making ClauseWise a powerful, research-friendly tool for both professionals and non-experts.

# GitHub Repository Link:

<https://github.com/MeruguSanthiPriya/CLAUSEWISEproject.git>