Software Requirements Specification (SRS) for AI-Powered Legal Document Analyzer

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) defines the requirements for an AI-Powered Legal Document Analyzer, a web-based platform that uses Natural Language Processing (NLP) to analyze legal documents, extract key clauses, identify risks, and provide actionable insights. The platform targets lawyers, law firms, and individuals, offering a scalable solution to streamline contract review and reduce manual effort. It supports monetization through subscriptions and B2B licensing, capitalizing on the growing legaltech market.

1.2 Scope

The platform provides:

- Automated analysis of legal documents (e.g., contracts, NDAs) to extract clauses, terms, and obligations.
- AI-driven risk assessment (e.g., identifying ambiguous clauses, non-compliance risks).
- User-friendly dashboards for document management, analysis results, and collaboration.
- Secure user authentication with role-based access (individuals, lawyers, law firm admins).
- Integration with external systems (e.g., cloud storage, legal databases).
- Monetization via freemium subscriptions and B2B law firm licensing.

Key Features:

- Document upload and parsing (PDF, Word, text).
- NLP-based clause extraction and risk scoring.
- Collaborative tools for team-based document review.
- Freemium model with premium analytics and API access.
- B2B integrations for law firm workflows and legal databases.
- Compliance with GDPR, CCPA, and legal data security standards.

Tech Stack:

- **Frontend**: React.js with Material-UI for responsive, intuitive interfaces.

- **Backend**: Python Django with Django REST Framework for API-driven services.
- **AI/ML**: spaCy for NLP tasks (NER, text parsing), Hugging Face Transformers for advanced text analysis, Scikit-learn for risk scoring.
- **Database**: PostgreSQL for relational data (users, documents, metadata).
- **Storage**: AWS S3 /Cloudinary for document storage, with encryption for security.
- **Authentication**: JWT for secure sessions, OAuth 2.0 for third-party logins (e.g., Google, Microsoft).
- **Analytics**: Google Analytics for user engagement, custom ML metrics for document analysis accuracy.

1.3 Definitions, Acronyms, and Abbreviations

- **NLP**: Natural Language Processing
- **NER**: Named Entity Recognition
- **JWT**: JSON Web Token
- **API**: Application Programming Interface
- **MVP**: Minimum Viable Product
- **B2B**: Business-to-Business
- **GDPR**: General Data Protection Regulation
- **CCPA**: California Consumer Privacy Act
- **UI/UX**: User Interface/User Experience
- **SRS**: Software Requirements Specification

1.4 References

- Django Documentation: https://docs.djangoproject.com/
- React Documentation: https://reactjs.org/
- spaCy Documentation: https://spacy.io/
- Hugging Face Transformers: https://huggingface.co/docs/transformers/
- AWS Documentation: https://aws.amazon.com/documentation/

1.5 Overview

This SRS outlines functional and non-functional requirements, system architecture, constraints, and assumptions for developers, stakeholders, and project managers. It ensures alignment on delivering a scalable, secure, and innovative legaltech solution.

2. Overall Description

2.1 Product Perspective

The AI-Powered Legal Document Analyzer is a standalone web application with potential for mobile extensions. It integrates with cloud storage (e.g., Google Drive, Dropbox) and legal databases (e.g., Westlaw via APIs) to enhance functionality. Its AI-driven approach differentiates it from competitors like DocuSign or Kira Systems by offering advanced NLP capabilities at a lower cost, targeting both individual users and law firms.

2.2 Product Functions

- **User Management**: Secure registration, login, and role-based access for individuals, lawyers, and admins.
- **Document Upload and Parsing**: Upload legal documents (PDF, Word) and extract text for analysis.
- **AI-Driven Analysis**: Extract clauses, identify risks, and summarize key terms using NLP.
- **Risk Assessment**: Score clauses for ambiguity, compliance risks, or unfavorable terms.
- **Collaboration Tools**: Annotate documents, share with team members, and track changes.
- **Monetization**: Freemium model with premium features (e.g., advanced analytics, API access) and B2B licensing.
- **Analytics Dashboard**: View document analysis metrics, user activity, and system performance.
- **Notifications**: Alerts for analysis completion, team comments, or subscription updates.

2.3 User Classes and Characteristics

- **Individual Users**: Upload personal contracts (e.g., leases), view simplified analyses. Need intuitive UI.
- **Lawyers**: Analyze complex contracts, collaborate with clients, access advanced NLP features. Require robust tools.
- **Law Firm Admins**: Manage users, subscriptions, and integrations with firm workflows. Need admin dashboards.
- **B2B Clients (Law Firms)**: Purchase bulk licenses, integrate with internal systems, access firm-wide analytics.

2.4 Operating Environment

- **Client-Side**: Modern web browsers (Chrome, Firefox, Safari) with mobile responsiveness.
- **Server-Side**: Django application hosted on AWS EC2 with PostgreSQL on AWS RDS.
- **AI/ML**: spaCy and Hugging Face models served via Django, with pre-trained models hosted on AWS ECS or your cloud service choice
- **Network**: Stable internet for document uploads and real-time collaboration.

2.5 Design and Implementation Constraints

- Compliance with GDPR/CCPA for data privacy and legal document security.
- Scalability to handle 5,000 concurrent users in the MVP phase.
- NLP model accuracy must exceed 85% for clause extraction and risk scoring.
- Budget constraints: Leverage open-source tools (Django, React, spaCy) and cost-effective AWS services.

2.6 Assumptions and Dependencies

- Users have access to legal documents in PDF, Word, or text formats.
- Third-party APIs (e.g., Stripe, Google OAuth) are reliable.
- Pre-trained NLP models (e.g., Hugging Face's BERT) are sufficient for MVP; custom training occurs post-launch.
- Law firms have APIs or manual processes for B2B integration.

3. System Features

3.1 User Authentication and Authorization

Description: Secure user registration, login, and role-based access control.
Priority: High
Functional Requirements:

- Users register with email/password or OAuth (Google, Microsoft).
- JWT-based authentication for secure API access.
- Role-based access: Individuals (view personal documents), Lawyers (analyze/collaborate), Admins (manage users/subscriptions).
- Password reset via email with secure token.
- Input: Email, password, or OAuth token.
- Output: JWT token, user profile data, or error message.
  Non-Functional Requirements:
- Response time: <1.5 seconds for login.
- Security: Encrypt passwords with bcrypt, HTTPS for all requests.

3.2 Document Upload and Parsing

Description: Users upload legal documents for AI analysis.
Priority: High
Functional Requirements:

- Support PDF, Word, and text file uploads (up to 50MB).
- Extract text using libraries like PyPDF2 or python-docx.
- Validate document format and integrity before processing.
- Input: Document file, user metadata.
- Output: Parsed text stored in AWS S3, metadata in PostgreSQL.
  Non-Functional Requirements:
- Upload time: <5 seconds for 10MB files.
- Storage: Encrypted storage in AWS S3 with access logs.

3.3 AI-Driven Document Analysis

Description: Extract clauses, summarize terms, and identify key entities using NLP.
Priority: High
Functional Requirements:

- Use spaCy for Named Entity Recognition (NER) to extract entities (e.g., parties, dates, amounts).
- Use Hugging Face Transformers (e.g., BERT) for clause classification and summarization.
- Generate summaries of key terms (e.g., obligations, deadlines).
- Input: Parsed document text.
- Output: JSON with extracted clauses, entities, and summaries.
  Non-Functional Requirements:
- Analysis time: <10 seconds for a 20-page document.
- Accuracy: >85% for NER and clause extraction in MVP.

3.4 Risk Assessment

Description: Identify and score risks in legal documents (e.g., ambiguous clauses, non-compliance).
Priority: High
Functional Requirements:

- Use Scikit-learn for risk scoring based on clause ambiguity, legal compliance, and industry standards.
- Highlight high-risk clauses (e.g., unclear termination conditions) with explanations.
- Provide risk scores (0-100) and mitigation suggestions.
- Input: Analyzed document data.
- Output: Risk report with scores and highlighted clauses.
  Non-Functional Requirements:
- Scoring time: <5 seconds per document.
- Explainability: Provide human-readable risk rationale.

3.5 Collaboration Tools

Description: Enable team-based document review and annotation.
Priority: Medium
Functional Requirements:

- Users annotate documents (e.g., comments, highlights) in real-time.
- Share documents with team members via secure links.
- Track changes and version history.
- Input: Annotations, share requests, version updates.
- Output: Updated document with annotations, version logs.
  Non-Functional Requirements:
- Real-time updates via WebSockets: <1 second latency.
- Version history stored for 90 days.

3.6 Monetization and Subscriptions

Description: Freemium model with premium analytics and B2B licensing.
Priority: High
Functional Requirements:

- Free tier: Basic document analysis (up to 3 documents/month).
- Premium tier: Unlimited analysis, advanced risk scoring, API access via Stripe payments.
- B2B: Bulk licensing for law firms with custom pricing and analytics dashboards.
- Input: Payment details, subscription plan, license requests.
- Output: Subscription confirmation, license details, or error.

  Non-Functional Requirements:
- Payment processing time: <4 seconds.
- Security: PCI compliance for payments, audit logs for B2B licenses.

3.7 Analytics Dashboard

Description: Provide insights on document analysis and user activity.
Priority: Medium
Functional Requirements:

- Individual dashboard: View analysis history, risk summaries.
- Lawyer dashboard: View client document metrics, collaboration stats.
- Admin dashboard: Platform-wide metrics (e.g., active users, revenue, model accuracy).
- Input: User role, document data.
- Output: Visual charts (React Chart.js), tabular data.

  Non-Functional Requirements:
- Dashboard load time: <2 seconds.
- Data refresh