Relational Knowledge Engineering Platform
Product Requirement Document[*]

Yongyuth Chuankhuntod[†] and Anirach Mingkhwan[‡]
Faculty of Industrial and Technology Management
King Mongkut's University of Technology North Bangkok

This document serves as a Product Requirement Document (PRD) for a bachelor's thesis project, not for submission to a conference or academic journal. It outlines the requirements and specifications for developing a Relational Knowledge Engineering Platform - a web-based tool designed to extract and visualize relationships between pieces of knowledge from uploaded documents.

Purpose: Bachelor's thesis project documentation and development guidance.

Scope: Complete product requirements for a knowledge engineering platform including document processing, relationship extraction, network visualization, and LLM-powered chat interface.

I. INTRODUCTION

The Relational Knowledge Engineering Platform is a web-based tool designed to help users extract and visualize relationships between pieces of knowledge contained in uploaded documents (e.g., PDFs). The platform processes text from these documents, generates a network graph to represent relationships, and provides interactive features for users to explore and manipulate the graph. Additionally, a chat interface powered by a Large Language Model (LLM) allows users to query their data, making it a powerful tool for knowledge discovery and management.

II. OBJECTIVE

The platform aims to:

- Enable users to upload text-containing files (e.g., PDFs) and automatically extract knowledge relationships.
- Present extracted relationships as an interactive network graph.
- Allow users to customize graph generation, interact with the graph (extract, add, delete elements), and summarize it based on metadata like text sentences in nodes.
- Provide a chat interface with an LLM for querying the graph and documents.
- Offer a user-friendly experience with account management and an engaging landing page.

III. TARGET AUDIENCE

- Researchers, analysts, and students needing to analyze relationships within textual data.
- Professionals in knowledge management, data science, or education seeking visual and interactive tools for text analysis.

IV. KEY FEATURES

A. Landing Page

Purpose: Introduce the platform and attract users.

Features:

- Overview of the platform's capabilities (e.g., knowledge extraction, graph visualization, LLM chat).
- Call-to-action buttons for signing up or logging in.
- Examples or use cases highlighting the platform's value.

B. Account Management (Own Account)

Purpose: Allow users to manage their personal settings.

Features:

- Profile management (e.g., name, email).
- Password reset or change functionality.
- User preferences (e.g., notification settings).

[*] Product Requirement Document
[†] yongyuth.chuankhuntod@email.kmutnb.ac.th
[‡] anirach.m@fitm.kmunt.b.ac.thh

C. Graph Viewer

Purpose: Display and interact with the network graph.

Features:

- Interactive graph visualization with zoom, pan, and node/edge selection.
- Display of node details (e.g., text sentences) and edge relationships.
- Tools to:
  - **Extract**: Highlight or isolate specific nodes/edges.
  - **Add**: Manually insert new nodes or relationships.
  - **Delete**: Remove nodes or edges.
- Summarize the graph based on metadata (e.g., key sentences or themes).

D. Chat with LLM

Purpose: Enable users to query their data conversationally.

Features:

- Chat interface for inputting questions about the graph or documents.
- LLM responses tailored to the user's uploaded content.
- Optional integration with the graph viewer (e.g., referencing specific nodes in responses).

E. File Upload and Graph Customization

Purpose: Handle document uploads and graph generation settings.

Features:

- Upload support for files like PDFs.
- Text extraction and relationship identification from uploaded documents.
- Customization options (e.g., select document sections, adjust relationship extraction parameters).
- Preview of the generated graph before finalizing.

V. TECHNICAL REQUIREMENTS

A. Frontend

- **Framework**: Next.ts (TypeScript) for a dynamic, interactive interface.
- **Graph Visualization**: Reagraph for rendering and interacting with the network graph.
- **Design**: Responsive layout compatible with desktop and mobile browsers.

B. Backend

- **Framework**: Go or Python for robust server-side logic.
- **NLP**: spaCy or transformers for entity and relationship extraction.

C. Database

- **Graph Database**: Neo4j graph database for storing and querying knowledge relationships.
- **Vector Database**: Qdrant for storing and querying document embeddings and context for LLM memory.
- **General Database**: MongoDB for general-purpose data storage (e.g., user accounts, settings, metadata).
- **Cache Database**: DragonflyDB for caching and pub/sub messaging.
- **Metadata**: Store node/edge details (e.g., text sentences) alongside graph structure.

D. File Storage

- **Object Storage**: MinIO for storing all uploaded files (PDFs, documents) with S3-compatible API.
- **File Management**: Secure file upload, storage, and retrieval with proper access controls.
- **Backup**: Automated backup and versioning of stored files.

E. LLM Integration

- **Provider**: API integration with OpenAI or Hugging Face for LLM functionality.
- **Context**: Pass graph data and document text to the LLM for contextual responses.

F. Security

- **Authentication**: JWT-based user authentication.
- **Data Protection**: HTTPS and encryption for uploaded files and user data.

G. Scalability

- **Infrastructure**: Cloud hosting (e.g., XVER) to support multiple users and large files.
- **Optimization**: Efficient graph generation and querying for performance.

VI. USER WORKFLOW

1. **Sign Up/Log In**: Users create an account or log in via the landing page.
2. **Upload File**: Users upload a PDF on the file upload page.
3. **Customize Graph**: Users adjust settings for relationship extraction and preview the graph.
4. **View Graph**: Users explore the network graph in the graph viewer, interacting with nodes and edges.
5. **Modify Graph**: Users extract, add, or delete elements and summarize the graph as needed.
6. **Chat with LLM**: Users ask questions about their data via the chat interface.
7. **Manage Account**: Users update their profile or settings on the account management page.

VII. SUCCESS CRITERIA

- **Functionality**: Accurate graph generation from uploaded files with full interactivity (extract/add/delete/summarize).
- **Usability**: Intuitive navigation and clear interfaces across all pages.
- **Performance**: Fast processing of documents and responsive graph interaction.
- **Insightfulness**: LLM provides relevant, accurate answers based on user data.

VIII. FUTURE CONSIDERATIONS

- Support for additional file formats (e.g., DOCX, TXT).
- Advanced graph analytics (e.g., clustering, centrality measures).
- Multi-user collaboration on shared graphs.

A. Second-level heading: Formatting

B. Document Structure

This document provides a comprehensive Product Requirement Document (PRD) for the Relational Knowledge Engineering Platform, structured to guide the development process from initial concept to implementation.

ACKNOWLEDGMENTS

This project is developed as part of a bachelor's thesis at King Mongkut's University of Technology North Bangkok, Faculty of Industrial and Technology Management.