ClassName: KnowledgeGraph [cleaning and extraction]	
Responsibilities	**Collaborators Modules**
1. Extract text from pdf using phases as marker(**extract_text_after_pharase** method) 2. Store extracted pdf text into result dict (**results** attribute) 3.Process multiple PDF files in a directory (**process_pdfs_in_directory** method) 4. Remove specific words from the extracted text (**remove_words_from_line**, **word_removal** method) 5.Convert data to JSON object (**to_json** method) 6.Convert JSON to dictionary (**to_dict** method)	**os, fitz, re, nltk, json** **LLM Recommender class** **ReportProcessor class** **Probability class**

ClassName: LLM Recommender [LLM QA]	
Responsibilities	**Collaborators Modules**
1. set Replicate api token (**set_api_token** method) 2. set language model (**set_model** method) 3.load and index data to vector store (**load_data** method) 4. create query engine to have QA with LLM (**create_query_engine** method) 5.Get index (**get_index** method) 6.Perform query on engine (**query** method)	**llama_index.core {Settings, VectorStoreIndex, SimpleDirectoryReader}** **llama_index.embeddings.huggingface{HuggingFaceEmbedding}** **llama_index.llms.replicate{Replicate}** **transformers{AutoTokenizer}** **os** **KnowledgeGraph class**

ClassName: ReportProcessor [ML clustering vector embeddings]	
Responsibilities	**Collaborators Modules**
1. Load JSON Data (**load_json** method) 2. generate text embeddings from entire JSON (**get_embedding** method) 3.Get sectional embedding given section name (**get_section_embedding** method) 4. Get all combined embeddings (**get_all_combined_embeddings** method) 5. Find optimal number of clusters using the silhouette score (**find_optimal_clusters** method) 6. generate cluster reports (**cluster_reports** method) 7. Save cluster reports to there repective paths (**save_cluster_reports** and **cluster_and_save_report** methods)	**json, os, sentence_tranformers{SentenceTransformer}, numpy** **sklearn.cluster{Kmeans, AgglomerativeClustering, DBSCAN}** **sklean.metrics[silhouette_score}** **Knowledge Graph class**

ClassName: IncidentCategorizer [regex pattern probability]	
Responsibilities	**Collaborators Modules**
1.Categorize incidents matching the input regex pattern features (**categorize_incidents** method) 2. Calculate probabilities summing all occurrence and distribute weight on each feature (**calculate_probabilites** method) 3.Filter and Print file incidents that fall into the feature probability thresholds (**filter_and_print** method)	**os, re** **Knowledge Graph class**