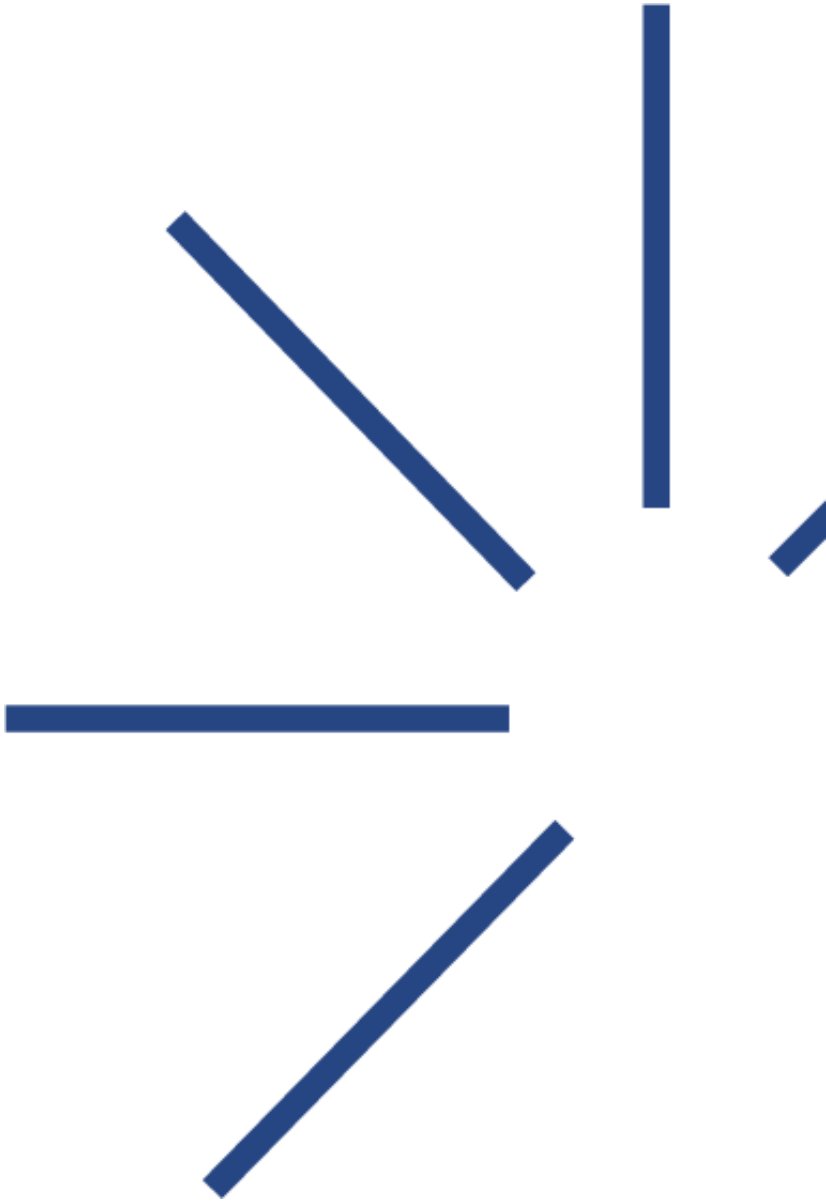
D15 – Repository design



Project funded by the European Union – Next GenerationEU through the Recovery and Resilience Plan of the Slovak Republic under project No. 09I05-03-V02-00049.

INTRODUCTION

The project **"Automation of Legal Text Analysis Based on Machine Learning"** (hereinafter referred to as **"ALTAML"**) represents a key initiative aimed at integrating innovative approaches in the field of data processing and subsequent analysis, specifically within the legal domain, which also includes the law of information and communication technologies. The objective of this project is to develop and validate effective methods for the automated analysis of legal texts using machine learning techniques. This includes, in particular, the development of tools that facilitate the processing and analysis of large volumes of data in the form of various legal documents, as well as the extraction of relevant information (attributes) from such documents, including the identification of key terms, references to other legal acts, and other attributes.

The ALTAML project thus aims to contribute to a more efficient access to legal information and to the acceleration of legal processes, thereby ensuring a higher degree of legal certainty and improving the accessibility of legal texts, the results of their analysis, and relevant legal information for both professionals and the general public.

As part of work package KPB3, deliverable **D15 – Repository Design** was prepared. This document focuses on the design of the system's data architecture for the automated processing, enrichment, and access to judicial decisions. It describes the proposed system components and their data repositories, including mechanisms for managing the processing pipeline, processing versioning, and the storage of enriched data.

The proposed design combines relational, document-oriented, and search databases to support the individual stages of legal text processing. Particular attention is paid to the design of repositories for linguistic text processing, legal reference extraction, and the identification of key legal concepts. The goal of the proposed architecture is to provide a scalable and extensible foundation for the further development of the system.

1. System Design and Associated Repositories

The proposed system represents a modular architecture designed for the automated processing, enrichment, and access to judicial decisions. An overall view of the proposed architecture and its repositories is illustrated in Figure 1.

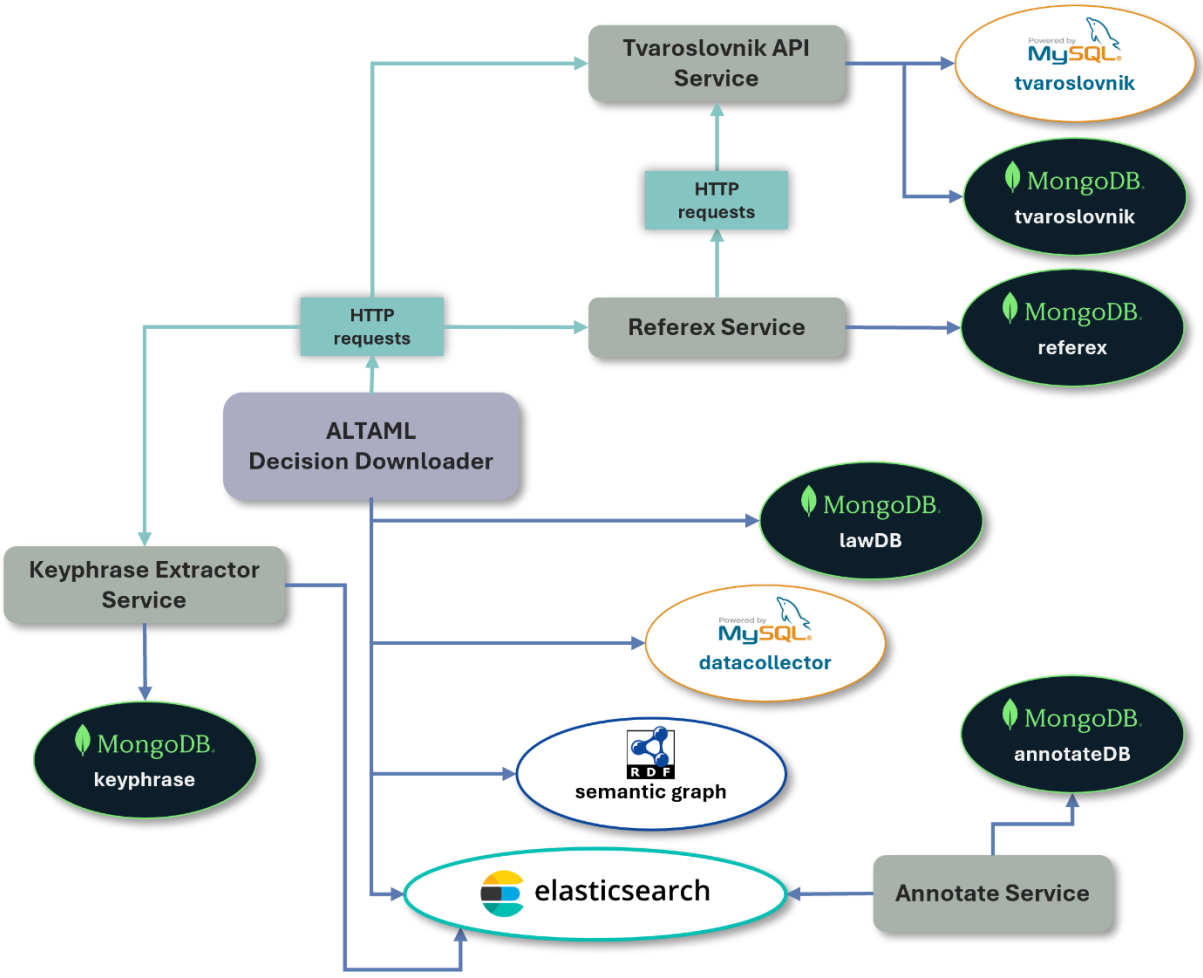


Figure 1 - Repository design

The core of the system is the ALTAML Decision Downloader component, which is intended to ensure the acquisition of decisions from external sources, their processing, versioning, and storage in appropriate data repositories.

Linguistic text processing is to be provided by the Tvaroslovník API Service, which offers lemmatization and grammatical information for the Slovak language.

The extraction of structured references to legal regulations and judicial decisions is to be performed by the standalone Referex Service, while the identification of key legal concepts is to be handled by the Keyphrase Extractor Service using semantic vector representations. The final search and analytical interface of the system is to be

provided by the Annotate Service, enabling search over judicial decisions and their annotation within a user-facing tool. The proposed architecture is based on clearly separated components and repositories, allowing for scalability, repeatable processing, and the gradual extension of the system.

2. ALTAML Decision Downloader Component

The ALTAML Decision Downloader is a component responsible for the automated downloading, processing, and enrichment of judicial decisions from external sources, primarily from the Ministry of Justice API (Slov-lex). It ensures decision version control, conversion of documents into textual form, and the sequential application of independent extraction methods. The output of this component consists of structured and enriched decisions prepared for storage.

The ALTAML Decision Downloader uses multiple repositories depending on the processing stage. **MySQL** serves as a control repository for recording decisions, their versions, and technical metadata. **MongoDB** is used to store intermediate and final enriched versions of decisions, including the history of extraction methods. **Elasticsearch** represents the final search repository intended for full-text, attribute-based, and semantic search over judicial decisions. In addition, the ALTAML Decision Downloader stores selected extracted relationships in a **semantic graph**, which does not contain the full texts of decisions but only structured entities and their mutual relationships.

MongoDB Database *lawDB*

The **lawDB** database serves as the primary storage for fully enriched decisions in the internal data model. A decision is stored in the database only after it has been completely processed—i.e., after PDF-to-text conversion and the execution of all extraction steps (e.g., legal regulations, references to other decisions, key legal concepts, and additional attributes). Fully enriched decisions are stored in the **decision_clean_raw** collection.

The database also stores information about the versions of the methods used, allowing multiple versions of the final object to exist for a single judicial decision depending on the extraction method versions applied. This approach enables reverting to earlier versions if a new extraction method proves to be faulty and allows new or modified extraction methods to be executed only on already fully processed decisions, with the results stored under a new version without rerunning the entire processing pipeline.

After all extraction and processing steps are completed and the results are stored, one specific version of the decisions is selected as the current version. Judicial decisions in

this version are then used by downstream systems, in particular for search via the Elasticsearch database and the Annotate service.

MySQL Database *datacollector*

The MySQL **datacollector** database serves as the control and reference database of the ALTAML Decision Downloader, with its primary role being the coordination of judicial decision processing. It does not store the content of decisions themselves, but rather the data required to determine what should be processed, in which version, and in what processing state a given decision currently resides.

A key element is the **raw_decision** table, which represents the central registry of all decisions identified in external sources. For each decision, it stores an internal identifier, information about the data source, the date of the last update in the source system, as well as technical details about the physical storage of the document (e.g., the path to the PDF file).
Based on this information, it is possible to distinguish new decisions from updated ones, prevent redundant processing, and accurately track the processing status.

MySQL also serves as a reference repository for shared and long-term valid data that are repeatedly used during processing. This mainly includes the **sud** (eng. court) table, containing normalized information about courts (court type, region, district), and the **data_source_schema** table, which stores both historical and current JSON response schemas from external APIs.
These data are stored deliberately to avoid repeatedly calling additional endpoints of external systems for each decision, thereby reducing the load on the Ministry of Justice servers and shortening processing time.

In addition, MySQL stores codebook data that are used in other parts of the project, primarily for filtering judicial decisions. These include the **povaha_rozhodnutia** (type of the decision), **oblast_pravnej_upravy** (area of law), **podoblast_pravnej_upravy** (sub-area of law), and **forma_rozhodnutia** (form of the decision) tables, which contain values corresponding to the meaning expressed by their table names.

Full-Text and Vector Search Repository Elasticsearch

Elasticsearch serves as the final search layer of the system over judicial decisions. Decisions from a selected version of the MongoDB decision_clean_raw collection are indexed in Elasticsearch, with each judicial decision represented as a single object.

This layer is not intended for storing the history or versions of decisions, but exclusively for their efficient retrieval and access. Elasticsearch is used for full-text search within decision texts, filtering based on structured and extracted attributes (e.g., court, case number, ECLI, decision date), and for advanced search scenarios.

During indexing, a custom analyzer with a Slovak Hunspell dictionary is used, which accounts for different morphological forms of words—an essential feature when working with Slovak legal texts.

3. Tvaroslovník (Word-Form Dictionary) API Service

The **Tvaroslovník API** is a REST service for processing the Slovak language. It is used for lemmatization (conversion of words to their base form) and/or for returning grammatical categories for a given word or a set of words. To perform these tasks, it uses two databases: **MongoDB** and **MySQL**.

MySQL Database *tvaroslovnik*

MySQL contains a single table, **lemma**, which forms the core of the lemmatization process. The *tvar (word form)* column stores the input (non-lemmatized) word form, the *lemma* column stores its base form, and the *pocet* (count) column represents the frequency of occurrence of the given base form in Slovak according to linguistic statistics. If multiple possible lemmas exist for a single word form, the one whose base form is more commonly used in Slovak (i.e., has a higher *pocet* (count) value) is preferred.

For performance reasons when processing longer texts, the table is optimized using a primary key, which enables fast lookup by word form.

MongoDB Database *tvaroslovnik*

Within the Tvaroslovník API, MongoDB is used to store data that are not directly part of the lemmatization table. The **kategorie** (categories) collection contains grammatical categories of words, where the document identifier (a document representing a single object within the collection) corresponds to a specific word form, and the stored data are used by an endpoint that returns grammatical information.

The **unlemmatized_words** collection records words that could not be lemmatized because they are not present in the MySQL **lemma** table. These words are subsequently reviewed and added by linguists, enabling their lemmatization in future processing.

4. Referex Service

Referex is a standalone service designed for the automated extraction of references from legal texts, in particular to Slovak legal regulations, previous judicial decisions, decisions of the European Court of Human Rights (ECtHR), the Court of Justice of the European Union (CJEU), and the Slovakian Office for Personal Data Protection.

This service uses MongoDB exclusively as its data repository. The following MongoDB collections are used within the Referex system to support the accurate identification of legal regulations and judicial decisions in legal texts.

MongoDB Database *referex*

The **zakony_aliases** collection contains full names of legal acts together with their identifiers and is used to detect matches of a law's name directly in the text of a legal document. For this purpose, it is complemented by the **dict_lemmatized_laws** collection, which contains the same law names in lemmatized form, enabling matching also in lemmatized decision texts.

Since judicial decisions often do not use full law names but only their abbreviations, the **law_aliases** collection is used. In this collection, the identifier represents a law abbreviation, and the *law_ids* attribute contains a list of law identifiers that may be denoted by that abbreviation. This collection is used to narrow the set of candidates when identifying a law in cases where only an abbreviation appears in the text. The **law_aliases_lemmatized** collection has the same content, but its identifiers are lemmatized, as they may also contain full words. During extraction, both collections are used, and the final result is selected based on the closest match to the legal text.

The **law_validities** collection contains documents where the identifier represents the law number (e.g., 300/2005 for the Criminal Code) and the *validities* attribute stores individual time intervals of validity of its versions. During legal reference extraction, this collection is used to assign the correct version of a law to the extracted reference.

The **codebook_courts** collection is used for extracting references to other Slovak judicial decisions. It contains court names and their types, derived from the court

codebook obtained from the ALTAML Decision Downloader, and enables reliable identification of the court mentioned in the decision text.

5. Keyphrase Extractor Service

The Keyphrase Extractor is a standalone service designed for the automated extraction of key legal concepts from judicial decisions. In computing key concepts, it uses the decision text itself as well as extracted references to legal regulations and other judicial decisions together with their texts.

To work with candidate concepts and to perform semantic comparison of vector representations, it uses the **MongoDB** and **Elasticsearch** databases.

MongoDB Database *keyphrase-extractor*

MongoDB serves as a reference repository for candidate legal concepts. The *candidate_terms* collection stores a set of candidate concepts divided into substantive and procedural categories, from which key concepts are selected for a specific judicial decision.

Vector Storage Elasticsearch

Elasticsearch is used as a vector database containing vector representations of the texts of legal regulations, judicial decisions, and their mutual relationships. Based on vector similarity comparisons, the semantic proximity of candidate concepts to the processed decision is evaluated, enabling the selection of the most relevant key legal concepts.

6. Annotate Service

The Annotate Service is a REST service that exposes endpoints for searching judicial decisions and at the same time provides full support for data annotation within an annotation tool.

Search is performed using Elasticsearch queries, which the Annotate service receives directly from the frontend annotation tool and executes over the index of judicial decisions.

Full-Text and Vector Search Repository Elasticsearch

Originally, the design considered developing a custom query language and a middleware layer for complex filtering of legal documents; however, this approach was replaced by the use of Elasticsearch's native query mechanism. This solution proved to be sufficient while also enabling the combination of traditional attribute-based and full-text search with semantic search based on vector representations stored directly in Elasticsearch.

MongoDB Database *annotate*

The MongoDB database is used for creating datasets of judicial decisions, managing annotation tasks, assigning tasks to annotators, and storing the annotations themselves.