

**Publisher**<http://jssidoi.org/esc/home>**USE OF ARTIFICIAL INTELLIGENCE FOR BUSINESS DECISION-MAKING*****Olga Cherednichenko ¹, Vladyslav Maliarenko ², Judita Táncošová ³, Ľubomír Nebeský ⁴**^{1,3,4} Bratislava University of Economics and Management, Bratislava, Slovakia² National Technical University "Kharkiv Polytechnic Institute", Kharkiv, UkraineE-mails: ¹ olga.cherednichenko@vsemba.sk; ² vladyslav.maliarenko@cs.khpi.edu.ua; ³ judita.tancosova@vsemba.sk; ⁴ lubomir.nebesky@vsemba.sk

Received 3 September 2025; accepted 27 January 2026; published 30 March 2026

Abstract. The Decision Model and Notation (DMN) standard provides a unified framework for the representation of business decisions, with decision tables constituting its central construct. Despite the advantages offered by the manual creation of DMN decision tables from textual descriptions in terms of transparency and analysis, the process is time-consuming and prone to errors. This results in a discrepancy between the textual rules and the formal structures required for automation. The proposed framework integrates Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG), facilitating the automated conversion of natural language descriptions into decision tables. The framework utilises a modular pipeline comprising schema injection, retrieval of relevant domain-specific data, prompt composition, generation, and validation. This design guarantees that the resulting decision tables comply with DMN's rigorous XML-based specification while also capturing the contextual meaning of the original rules. A significant challenge in relying exclusively on LLMs is their proclivity to hallucinate or deviate from the required format. The integration of RAG has been demonstrated to mitigate these issues by grounding generation in reliable sources, enhancing both factual accuracy and structural consistency. The experimental results demonstrate that the LLM-RAG configuration significantly enhances the precision and robustness of generated DMN decision models, producing valid and contextually appropriate outputs with minimal human intervention. The advancement of automation in the field of decision modelling has the potential to enhance the efficiency, transparency and adaptability of decision-making processes. The findings indicate a pathway towards scalable, explainable, and verifiable decision automation across diverse business domains.

Keywords: ICT for business; Decision Model and Notation (DMN); Decision-making; Large Language Model; Natural Language Processing; Retrieval-Augmented Generation (RAG); Artificial Intelligence (AI)

Reference to this paper should be made as follows: Cherednichenko, O., Maliarenko, V., Táncošová, J., Nebeský, D. 2026. Use of artificial intelligence for business decision-making. *Entrepreneurship and Sustainability Issues*, 13(3), 345-359. <http://doi.org/10.9770/r6749942424>

JEL Classifications: C02, C4, D81, M19

Additional disciplines: information and communication; informatics

1. Introduction

In today's data-driven economy, organisations increasingly rely on formalised decision models to ensure consistent, transparent, and adaptable operations (Mugunzva & Manchidi, 2024; Gavaza, 2025; Španková & Sochuľáková, 2025). The Decision Model and Notation (DMN) standard, developed by the Object Management Group (OMG), has emerged as a leading framework for modelling complex decision logic using structured

* This research was funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I03-03-V01-00078

decision tables (Calvanese, Dumas, & Maggi, 2018). These tables allow business users and analysts to formalise rule-based decisions in a format that is both machine-readable and human-interpretable. However, despite the growing adoption of DMN, the process of manually creating decision tables from natural language rules remains labour-intensive, error-prone, and inaccessible to non-experts.

A decision table is one way to represent the decision logic associated with a Decision Requirements Diagram artefact. The table includes all necessary inputs for determining the outputs for every possible combination of inputs. Decision tables, along with hierarchical structures of decision tables, are a well-established method for representing decision logic (Arco et al., 2021; Vandevelde, Aerts & Vennekens, 2023; van Woensel & Motie, 2024).

The rapid advancement of Large Language Models (LLMs), such as GPT-4 and Claude 3.5, has opened new possibilities for automating decision model generation. These models are capable of interpreting natural language instructions, identifying decision variables, and producing structured outputs. Recent research has demonstrated that LLMs can generate basic DMN tables from unstructured text, but challenges persist in ensuring rule completeness, exclusivity, and logical consistency. Furthermore, standalone LLMs often suffer from hallucinations, especially when critical domain-specific knowledge is missing. To address these limitations, the integration of Retrieval-Augmented Generation (RAG) has been proposed. RAG improves generation accuracy by allowing the model to query external knowledge bases during inference, thereby grounding outputs in relevant, up-to-date information. Combined with schema-aware prompting and post-generation validation tools, this approach significantly enhances the reliability of LLM-generated DMN tables.

This research investigates how LLMs and RAG can be jointly leveraged to automate the generation of DMN decision tables from natural language. We analyse the structure of DMN models, identify key bottlenecks in their manual creation, and propose a framework that enables automated generation, validation, and deployment of decision logic. Through empirical testing and literature-driven analysis, we aim to reduce the complexity of DMN modelling and move toward scalable, LLM-driven decision automation.

The primary aim of this research is to develop and evaluate a modular framework for the automated transformation of natural language business rules into DMN decision tables by leveraging LLMs enhanced with RAG. The study seeks to address the challenges of schema compliance, logical consistency, and factual grounding in the automatic generation of decision logic.

To achieve this aim, the study pursues the following objectives: (1) to analyse existing approaches to DMN-based decision modelling and identify the key limitations of manual table construction; (2) to explore the capabilities and constraints of LLMs in generating structured DMN models from textual descriptions; (3) to design a hybrid architecture integrating schema injection, retrieval, prompt composition, generation, and validation components; (4) to conduct empirical testing comparing the performance of LLM-only and LLM+RAG configurations in a realistic decision-making scenario; and (5) to evaluate the impact of RAG on the accuracy, completeness, and reliability of automatically generated DMN tables. Accordingly, this research is guided by the following questions:

- (1) To what extent does the integration of RAG improve the accuracy and completeness of DMN decision tables generated by LLMs?
- (2) How do schema injection and retrieval mechanisms contribute to reducing structural and logical inconsistencies in automated model generation?
- (3) What is the potential of the proposed framework to enable scalable, explainable, and verifiable decision automation across business domains?

2. Literature Review

The DMN model is specifically designed for business professionals with the decision table as its core construct. Key elements within DMN tables are built upon FEEL, the standard DMN language, which is based on a simple data model and uses three-valued logic (true, false, null). We can highlight several key elements. An Information Item Name identifies what the table represents, typically linked to the decision logic. The table includes input

clauses made up of simple input expressions, such as names or conditions, and may define allowed values. These inputs serve as the basis for the rules within the table. Outputs are defined through the output clauses, each containing a name and specifying allowed output values. The decision table's rules collectively establish how particular combinations of input values lead to specific outputs, forming a clear, structured approach to decision-making. The rules can be presented in rows or columns, depending on the table's orientation. There are normally several rules, which do not overlap. The hit policy indicator is required to recognise the table type and understand the decision logic.

Empirical applications, particularly in complex domains such as healthcare, underscore the practical utility of DMN in automating decision-intensive processes. By enabling decisions to be modelled and evaluated independently, DMN improves decision quality, supports transparency, and enhances organisational adaptability to frequent changes in business rules. Thus, DMN decision tables are an essential tool for advancing automated decision-making processes in diverse operational contexts (Corea et al., 2024).

The goal of DMN is to provide the necessary constructs for decision modelling, allowing to be automated. With the growing adoption of DMN decision tables, there is an increasing need for a comprehensive analysis of these tables. Findings indicate that model complexity decreases when processes are distilled to their essence, avoiding hard-coded decision logic (Funk, 2023). The introduction of the DMN standard has sparked a variety of technical and empirical studies focused on decision models created using DMN (Fernandez et al., 2023).

The integration of DMN with BPMN represents a paradigm shift in how organisations manage and automate their workflows. In practice, BPMN activities that require decision-making, such as "business rule tasks," can reference DMN decision models. These references link the inputs and outputs of a BPMN process to the decision-making logic encoded in DMN tables. For instance, in a procurement process, a BPMN task to approve purchase orders can utilise a DMN decision table to evaluate the order's compliance with business policies and budget constraints. This linkage ensures that processes remain consistent, and decisions are executed uniformly, regardless of the process context (Etikala, 2021). By providing a standardised framework for integrating processes and decisions, the BPMN-DMN synergy enables organisations to streamline their operations and adapt more effectively to a dynamic business landscape (Kanyuka & Mahfoud, 2024).

The automation of DMN table generation using LLMs represents a significant advancement in decision automation. Traditional DMN modelling relies on manual rule definition, requiring domain expertise and time investment. The integration of LLMs into this process offers the potential to streamline DMN table generation by transforming natural language descriptions into structured decision logic (Boonmepit & Suwannasart, 2019). While LLMs have demonstrated strong capabilities in structured data generation, several key challenges remain when applying them to DMN modelling.

Ensuring schema compliance is one of the primary challenges in LLM-driven DMN table generation. Since DMN tables are implemented in an XML-based format, every generated table must adhere to predefined input clauses, output clauses, hit policies, and decision rules to be executable in DMN engines. However, LLMs, when generating structured XML outputs, often introduce formatting inconsistencies, such as missing attributes, incorrect nesting, or misaligned column structures. These errors primarily arise due to the probabilistic nature of LLMs, which prioritise fluency over strict schema adherence, leading to schema deviations that require manual correction (Estrada-Torres et al., 2024; Tam et al., 2024). One proposed solution is schema-aware prompting (Corea et al., 2019), which explicitly includes XML-based DMN templates in the prompt to enforce structural integrity. Additionally, format-restricting instructions and structured schema constraints have been shown to improve LLM compliance with XML-based decision tables. However, research highlights a trade-off between schema rigidity and reasoning performance, as excessively constrained generation can degrade the model's ability to produce contextually appropriate outputs.

LLM-generated DMN tables must maintain logical consistency to ensure that rules do not contradict each other, cover all necessary conditions, and align with the specified decision logic. However, due to the stochastic nature of LLM outputs, generated decision tables often contain conflicting rules, redundant conditions, or missing dependencies, leading to inconsistencies in decision automation (Goossens et al., 2022). These logical

inconsistencies are particularly problematic in domains where decision logic involves multi-variable dependencies and cascading rule applications. To address this, post-generation validation frameworks have been explored, enabling automated rule verification and conflict detection (Shorten et al., 2024). Some approaches integrate constraint-based verification systems, where generated DMN tables are analysed against predefined business logic constraints to ensure compliance. Additionally, iterative refinement techniques, where LLM-generated tables undergo secondary validation using rule-checking models, have demonstrated improvements in logical accuracy (Berkovitch et al., 2025).

A major limitation of LLM-generated DMN tables is hallucination, where the model produces non-existent or logically incorrect decision rule (Calvanese et al., 2018). Unlike traditional errors, hallucinations appear as plausible but fabricated rules, often leading to incorrect decision-making outputs. These errors occur because LLMs generate content based on statistical inference rather than direct retrieval of domain-specific knowledge, making them susceptible to introducing unsupported rules or fabricated dependencies (Li et al., 2022). To mitigate hallucination, retrieval-augmented generation has been proposed as an effective strategy, allowing LLMs to retrieve validated decision rules before generating outputs (Goossens et al., 2023). This method significantly reduces the likelihood of hallucinations by anchoring model outputs to pre-existing, verified decision logic. Additionally, schema-aware constraint injection has proven effective in minimising hallucinated rules in DMN tables (Tavantzis et al., 2023).

Complex formula generation is essential for DMN automation, as many decision models require mathematical computations and logical expressions rather than simple rule-based conditions (Bork et al., 2023). LLMs must not only generate syntactically correct formulas but also ensure they align with Friendly Enough Expression Language, the standard for DMN expressions (Leewis et al., 2024). However, maintaining both syntactic and logical correctness remains a challenge, as LLMs often introduce errors such as invalid syntax, missing dependencies, or logically inconsistent calculations (Liu et al., 2024).

Additionally, validation mechanisms play a crucial role in ensuring correctness (Goossens et al., 2023). Syntactic validation checks generated formulas against FEEL grammar rules, preventing syntax errors, while semantic validation ensures that computed outputs align with expected decision outcomes. By integrating these approaches, LLM-driven DMN automation can achieve more reliable formula generation, reducing the need for manual corrections and improving decision-making accuracy in real-world applications. Ensuring the correctness of LLM-generated DMN tables requires robust validation mechanisms to detect syntax errors, logical inconsistencies, and missing conditions (Liu et al., 2024). While manual validation is an option, it is inefficient for large-scale decision automation (Jeong, 2023). Research suggests that automated rule validation frameworks can significantly improve the reliability of DMN tables by systematically checking for structural completeness and logical soundness.

One approach to validation is formal verification, where decision tables are analysed using geometric representations and dependency graphs to detect overlapping or missing rules. Another strategy involves test case validation, where generated DMN tables are executed against sample datasets to verify that outputs align with expected decision outcomes (Funk, 2023). Combining these techniques with machine learning-based anomaly detection further enhances the accuracy and reliability of LLM-generated decision models.

Large Language Models have emerged as a transformative tool for automating the generation of DMN tables, enabling the structured representation of decision logic directly from natural language inputs (Bork et al., 2023). Traditional DMN modelling requires manual definition of input conditions, rules, and outputs, which can be both time-consuming and prone to inconsistencies. By leveraging LLMs, organisations can streamline this process, reducing manual effort while improving scalability and accessibility for non-technical users. However, despite their promise, LLMs face several technical challenges when applied to DMN table generation, including schema compliance, logical consistency, hallucination control, and complex formula generation. One of the primary obstacles in LLM-based DMN automation is schema adherence, as decision tables must conform to a strict XML-based format with predefined structures. Studies show that when LLMs generate structured XML or JSON outputs, they often struggle to maintain correct formatting, leading to missing attributes, incorrect clause definitions, or invalid hit policy assignments (Salinas & Morstatter, 2024). These issues arise because

LLMs generate text probabilistically rather than following a rigid schema. Approaches such as schema-aware prompting and format-restricting instructions have been explored to mitigate these errors, ensuring better compliance with DMN standards (Etinger et al., 2019).

Another critical challenge is logical consistency, as DMN decision tables require non-contradictory, exhaustive rule coverage. LLMs, however, tend to introduce redundant or conflicting rules when generating decision logic, particularly in complex tables that require interdependent rules. Ensuring that generated DMN tables maintain logical soundness often requires post-generation verification through automated validation frameworks. Some research suggests integrating rule-based validation mechanisms that can automatically detect and resolve conflicts, improving the reliability of LLM-generated decision models (Goossens et al., 2023; Woensel & Motie, 2024).

To enhance decision accuracy and reduce hallucination, retrieval-augmented generation has been proposed as a method to ensure LLM-generated DMN tables align with verified decision logic (Calvanese et al., 2018). Instead of relying solely on generative inference, LLMs retrieve validated decision rules from existing datasets before generating outputs, reducing the risk of fabricated or inconsistent rules. Experiments indicate that integrating structured retrieval with generative outputs significantly improves the factual consistency of generated DMN tables, especially in domains where regulatory or business constraints dictate decision structures.

Beyond retrieval-based improvements, formula generation within DMN tables remains a key limitation for LLMs. Many decision models require computed attributes, such as calculating a credit score based on financial parameters or determining eligibility based on derived conditions (Leewis et al., 2024). LLMs, when prompted to generate formulas, frequently produce syntactically valid but logically flawed expressions. Methods such as multi-step decomposition, where models generate formulas iteratively rather than in a single pass, have shown promise in improving accuracy (Liu et al., 2024).

While these advancements improve the reliability of LLM-driven DMN generation, further challenges remain in scalability, adaptation to domain-specific constraints, and the ability to refine decision logic dynamically. Future research should focus on embedding rule validation directly into the generation pipeline, enabling LLMs to detect errors in real-time rather than relying solely on post-processing corrections (Jeong, 2023). Additionally, enhancing adaptive schema enforcement techniques and domain-specific fine-tuning will be critical for making LLM-driven DMN automation robust enough for real-world applications.

3. Background

This paper presents a series of case studies that illustrate the practical applications of DMN tables in automated decision-making processes. Each scenario highlights how DMN enables organisations to address specific challenges, such as streamlining operations, improving decision quality, and adapting to changing business rules. These examples demonstrate the versatility and effectiveness of DMN in diverse industries. By examining these case studies, we aim to showcase the potential of DMN tables to revolutionise decision-making processes, offering insights into their implementation and impact. The scenarios presented not only provide a deeper understanding of DMN's capabilities but also underline its importance in achieving scalable, efficient, and accurate decision automation.

Scenario 1. Automated University Admission Screening Using DMN Decision Tables

In this case study, we explore the application of DMN decision tables in automating the university admission process to filter applicants based on their academic performance. The DMN table evaluates students' qualifications based on their declared department and scores in specific subjects such as Math, Physics, and English, along with an optional criterion like a motivational letter score. The automation process leverages the "Automatic Decision" column to determine whether a student's application should be accepted, declined, or requires further clarification (Figure 1).

The workflow begins when a student submits their application, specifying the desired department and the scores achieved in relevant subjects. The system, using the DMN table, cross-references these scores against

predefined criteria for each department. For instance, a student applying to the "Computer Science" department must meet a minimum score in Math or Physics to be considered, as shown in the rules of the decision table. Based on the table's calculations, an automatic decision is made, and the student is notified accordingly. This automated decision-making significantly reduces the workload of the admissions committee by eliminating the need to manually review applications that fail to meet minimum requirements or clearly qualify based on high scores. Students who do not meet the standards receive a rejection notification ("DECLINE"), while those who qualify are informed of the university's interest ("ACCEPT"). Applications requiring additional information, such as borderline cases or incomplete submissions, may trigger a request for clarification ("CLARIFY").

Students Acceptance Criteria Hit policy: First							
	When	And	And	And	And	Then	
	Department	Math	Physics	English	Geography	Automatic Decision	
	string	integer	integer	integer	integer	string	
1	"Computer Science"	< 138	-	-	-	"DECLINE"	
2	"Computer Science"	> 180	> 170	-	-	"ACCEPT"	
3	"Computer Science"	> 180	-	> 170	-	"ACCEPT"	
4	"Physics and Astronomy"	-	< 145	-	-	"DECLINE"	
5	"Physics and Astronomy"	> 160	> 185	-	-	"ACCEPT"	
6	"Physics and Astronomy"	-	> 185	-	> 170	"ACCEPT"	
7	-	-	-	-	-	"CLARIFY"	
+	-	-	-	-	-	-	BPMN.io

Figure 1. Students Acceptance Criteria Table

Source: Own work

The implementation of DMN decision tables in this context demonstrates the potential to streamline university admission processes by focusing committee resources on complex cases that require human judgment. This approach not only enhances efficiency but also ensures consistency in decision-making. By automating routine evaluations and focusing on candidates who meet or exceed criteria, universities can maintain a fair and transparent selection process while effectively managing high application volumes.

Scenario 2. Automated Gift Selection for Customers

In this case study, we explore the application of DMN decision tables in automating the process of selecting gifts for customers based on predefined criteria such as birthdays, holidays, and the value of their previous purchases. The decision table is used to evaluate a customer's eligibility for a gift, making the process quicker and more efficient while ensuring consistency in how customers are rewarded (Figure 2).

Gift Selection for Customers Hit policy: First							
	When	And	And	And	Then	And	
	Has Birthday Today	Is Holiday Today	Last Purchase Category	Last Purchase Price	Coupon Category	Decision	
	boolean	boolean	string	decimal	string		
1	true	false	"Electronics"	> 1000	"Celebrate the client!"	"10% discount on accessories"	
2	true	true	"Any"	> 750	"Holiday appreciation"	"Free shipping on next purchase"	
3	false	true	"Clothing"	> 1000	"Bring back the client"	"10% discount on all clothing"	
4	false	true	-	-	"Remind the client"	"Special offer for the first purchase"	
+	-	-	-	-	-	-	BPMN.io

Figure 2. Gift Selection for Customers Table

Source: Own work

The process begins when their birthday or a holiday occurs. The system checks the customer's data, including the type of their last purchase, their spending amount, and whether today is a special day like their birthday or a recognised holiday. Using the DMN table, the system then determines which gift or promotional offer to provide based on the combination of conditions. After the table processes the input data, the "Decision" column provides an automatic decision: the customer either receives a special offer or is reminded about an exclusive promotion. For instance, if it's the customer's birthday and they purchased electronics worth over \$1000, they

might receive a 10% discount on accessories. If it's a holiday and the last purchase was any item over \$750, they might be given free shipping on their next purchase.

This automated decision-making significantly reduces the workload on marketing teams, as it eliminates the need for manual gift selection. Customers who meet the criteria automatically receive a notification of their gift or offer, ensuring that the process is fast, efficient, and personalised. By automating these decisions, the business can focus resources on more complex tasks, while ensuring that customers who qualify for rewards based on simple criteria are quickly and consistently recognised.

Scenario 3. Alternative Program Recommendations for University Applicants

In this case study, we examine how DMN decision tables can automate the process of recommending multiple alternative academic programs to university applicants who do not meet the criteria for their first-choice program. This approach leverages the capability of DMN tables to generate multiple results, offering students a range of suitable options based on their academic profile and additional qualifications.

When a student submits an application, including their preferred program, academic scores in relevant subjects, and optional data such as motivational letter scores or extracurricular achievements, the system uses a DMN decision table to evaluate their eligibility across various programs. Unlike systems that return a single decision, the DMN table identifies all programs the student qualifies for and ranks them based on fit (Figure 3).

Alternative Program Recommendations for University Applicants						
				Hit policy:	Rule order	
When	And	And	And	Then	And	
First-Choice Program	Math Score	English Score	Extracurriculars	Recommended Programs	Reasoning	
string	integer	integer	string	string	string	
1 "Computer Science"	< 160	-	"Tech Club"	["Information Technology", "Applied Computing"]	"Lower math requirement, strong extracurricular profile in technology."	
2 "Engineering"	< 140	> 170	-	["Digital Media", "General Studies"]	"High English score aligns with creative and general education programs."	
3 "Computer Science"	< 160	-	"Robotics Club"	["Applied Computing"]	"Relevant extracurricular activities compensate for lower Math grades."	
4 -	< 140	< 140	-	["General Studies"]	"Does not meet specific thresholds, recommended for a broad foundational program."	
5 "Computer Science"	> 165	> 185	-	["Information Technology", "Digital Media"]	"Strong English score and moderate Math performance suggest technical programs."	
+	-	-	-	-	-	BPMN.io

Figure 3. Alternative Program Recommendations for University Applicants Table

Source: Own work

For instance, a student applying for "Computer Science" but failing to meet the Math requirement might receive recommendations for both "Information Technology" and "Applied Computing," especially if their extracurricular activities align with these programs. Similarly, a student with strong English skills but lower Science scores could be offered both "Digital Media" and "General Studies," giving them flexibility in choosing an academic path. This multi-result capability allows universities to provide applicants with personalised options while reducing manual evaluation efforts. By presenting a comprehensive list of alternative programs, along with explanations for each recommendation and clear next steps, the system ensures that students feel valued and supported. This approach demonstrates the flexibility and power of DMN decision tables in automating complex, multi-outcome decision-making processes.

In this scenario, the DMN decision table operates under the "Hit Policy: Rule Order", which ensures that all matching rules are evaluated, and their outputs are provided in the order they appear in the table. This approach is crucial for scenarios like alternative program recommendations, where the objective is to generate multiple results rather than a single decision. The "Rule Order" policy prioritises rules sequentially based on their placement in the table, allowing the system to rank recommended programs effectively. For instance, if a student qualifies for both "Information Technology" and "Applied Computing," the system lists them in the order specified in the decision table, ensuring transparency and alignment with the institution's priorities. This policy ensures that students receive a structured and prioritised list of alternatives, enhancing their ability to make informed choices while maintaining consistency in decision-making.

4. Research Objective and Methodology

While LLMs demonstrate promising capabilities in generating structured DMN tables from natural language input, they often fall short in scenarios requiring domain-specific reasoning, derived attributes, or strict adherence to schema constraints. These limitations become especially critical in real-world applications where the correctness and completeness of decision logic directly affect operational outcomes.

The goal of our research was to evaluate whether integrating RAG can significantly improve the quality and accuracy of DMN table generation in such challenging scenarios. Specifically, we focused on a use case where the required decision attribute – Body Mass Index (BMI) is not explicitly present in the input data but must be calculated from available fields (i.e., weight and height). This setup tests the model's ability to reason over available context and apply transformation logic as part of DMN table construction.

By comparing the performance of a pure LLM pipeline against an LLM & RAG approach, we aim to answer two questions:

- Can RAG help the model recover relevant transformation logic and apply it correctly during DMN generation?
- Does this integration improve the validation pass rate when DMN output is tested against a domain-specific rule engine?

This research thus provides a focused assessment of schema injection and derived logic reasoning two core challenges in the automation of DMN modelling and serves as a benchmark for the effectiveness of retrieval-based enhancement in structured code generation.

To evaluate the impact of RAG on the quality of DMN table generation, we designed a controlled experiment that simulates a real-world decision-making scenario. The research was centered around generating a decision table for BMI-level classification, where the target attribute (BMI level) must be inferred from other available data attributes – namely, weight, height, and gender. This setup challenges the model to not only generate syntactically valid DMN XML but also apply correct domain-specific logic for computing derived values.

We tested two approaches: a baseline using an LLM-only prompt and an enhanced variant where the same prompt was supplemented with a retrieval context (LLM & RAG). Both versions were evaluated by generating DMN tables from the same natural language instruction and validating them against a predefined set of 100 structured input objects. The goal was to assess whether the presence of retrieved background knowledge would improve rule completeness, accuracy of derived logic, and schema conformity. The experimental environment included automated validation using a DMN engine, allowing us to systematically quantify the success or failure of each generated table. This setup enabled a reproducible comparison of how well each approach handles non-trivial transformation logic and adheres to the structural requirements of DMN.

Input Data

The input for the experiment consisted of two core components: a natural language prompt describing the decision logic, and a set of structured test objects simulating real-world entity data. These inputs were designed to reflect realistic use cases in which decision logic must be inferred from partial or derived attributes rather than directly declared fields. The natural language prompt explained how to calculate a person's Body Mass Index based on their weight and height, and how to classify the resulting value into a corresponding BMI-level. The classification rules were gender-specific and covered overlapping and edge-case conditions. This prompt required the model to generate a decision table that includes an internal formula ($BMI = \text{weight} / (\text{height} * \text{height})$) and conditional logic that adapts based on both numerical ranges and categorical gender values.

The entity objects were provided in JSON format and included attributes such as id (integer), first and last names (string), gender (string), weight (double) and height (double).

Notably, these objects did not contain an explicit BMI value, thus requiring the model to understand that BMI must be computed dynamically before applying classification logic. A total of 100 mock entities were generated to cover a variety of valid and boundary-case scenarios (e.g., underweight, normal, overweight, obese), ensuring broad test coverage for each rule in the decision table. This design enabled testing not only for the generation of structurally correct DMN tables, but also for the model's ability to reason over entity properties and synthesise derived attributes when necessary. The test scenario was constructed to evaluate the model's ability to generate a complete and executable DMN decision table from natural language, especially when the required decision logic involves derived attributes. Specifically, the task involved classifying individuals into BMI-level categories based on computed Body Mass Index (BMI), which is not directly available in the input data. Figure 4 demonstrates the input sent to the model to generate DMN Table based on the rules.

*The BMI of a person can be calculated based on weight in kgs and length of a person in meters by using the following formula: $\text{weight} / (\text{height} * \text{height})$.*

If the BMI value is above 30, then the BMI-level is considered "obese".

If you are a male and the BMI-value is under 18.5 then the BMI-level is "severely underweight" and if you are female, then you are considered "underweight" with the same bmi-value.

If the BMI-value is between 18.5 and 25, then the BMI-level is "underweight" for a male and "normal" for a female.

Lastly, if the BMI-value is between 25 and 30 and you are a male then the BMI-level is "normal" but if you are a female then BMI-level is "overweight".

Figure 4. Test Input Data

Source: Own work

This scenario was executed across 100 test cases covering all possible branches of the logic. The experiment was repeated under two conditions: with LLM-only generation and with RAG-enhanced generation, allowing direct comparison of correctness and robustness across both settings.

Validation Strategy

To assess the correctness and robustness of the generated DMN decision tables, we implemented an automated validation pipeline based on rule execution over a predefined test dataset. This strategy allowed us to evaluate not only the structural integrity of the generated DMN XML code but also its ability to correctly classify input objects according to the expected logic.

The validation process involved the following steps:

1. **DMN Parsing and Deployment.** Each generated DMN table (in XML format) was parsed and deployed to a DMN engine capable of executing decision logic. Tables that failed to parse due to structural errors were marked as invalid immediately.
2. **Test Data Execution.** A set of 100 structured test objects was prepared in JSON format, each containing weight, length, and gender attributes. These inputs were executed against the deployed DMN table to compute the output column (BMI-level) for each case.
3. **Expected Output Comparison.** For each input object, the correct BMI value was precomputed, and the expected BMI-level classification was determined using the ground truth logic described in the prompt. The engine output was then compared against the expected result.
4. **Result Classification.** Each experiment run was classified as Passed if all 100 objects produced correct classifications and the DMN file was syntactically valid. Any mismatch in decision output or structural failure was marked as Failed.
5. **Metrics Reported.**
 - Pass Rate. Number of fully successful experiments divided by total attempts.

- Failure Reasons. Categorised into structural errors (invalid XML), logical errors (incorrect rule application), or missing decision branches.

This validation strategy ensured that we could rigorously test whether the LLM understood the full semantics of the decision logic, handled derived attributes correctly, and generated DMN tables suitable for deployment in decision engines. Comparing results from both LLM and LLM & RAG runs enabled a quantitative evaluation of the impact of retrieval-based augmentation.

5. Results and discussion

The research was conducted under two configurations: using an LLM-only pipeline and using an LLM enhanced with Retrieval-Augmented Generation (LLM & RAG). For each configuration, 100 independent test runs were executed using the same natural language prompt and input schema. The goal was to compare the ability of each approach to produce a valid and correct DMN decision table that passes automated validation against 100 structured input objects. The results clearly demonstrate the performance improvement when retrieval support is added in Figure 5.

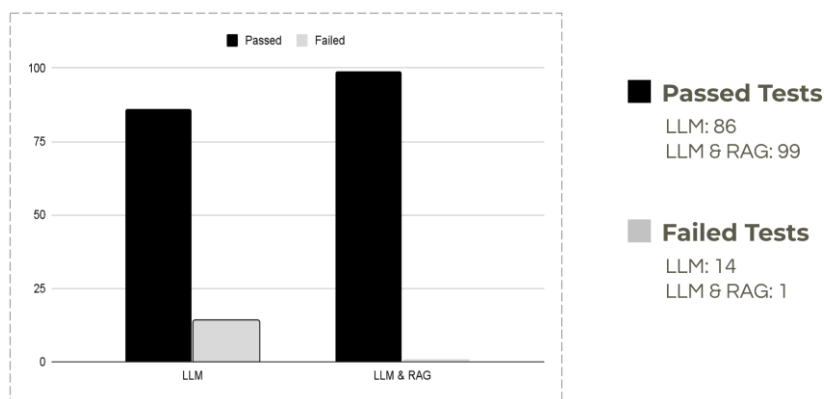


Figure 5. Test Results
Source: Own work

In the LLM-only setup, 14% of the generated DMN tables failed validation. Failures were primarily caused by:

- Incorrect or incomplete handling of BMI calculation logic.
- Missing classification branches (e.g., overlapping conditions or non-mutually-exclusive rules).
- Occasional syntactic issues in the generated XML structure.

By contrast, the LLM & RAG configuration achieved a 99% success rate. The addition of domain-specific retrieval (in this case, relevant logic for BMI calculation and classification) provided critical grounding information that allowed the model to:

- Correctly include BMI calculation as part of the decision logic.
- Apply all conditional branches with the appropriate gender-specific logic.
- Avoid hallucinated rules or structural omissions.

These results confirm that RAG significantly improves both logical completeness and structural correctness of the generated DMN tables. Moreover, the experiment highlights the value of retrieval, especially in cases where decision attributes are derived rather than directly present in the schema.

The experimental results highlight several important implications for the field of automated DMN table generation. First, they confirm that while LLMs are capable of producing structured decision logic from natural language, their reliability in non-trivial cases is limited without external knowledge grounding. The LLM-only configuration performed reasonably well on simpler cases but consistently failed in situations where reasoning over implicit schema relationships was required. This finding is consistent with earlier observations in the

literature that LLMs tend to hallucinate missing logic or omit edge conditions when prompted without access to structured context.

In contrast, integrating Retrieval-Augmented Generation significantly improved the model's output by providing it with relevant procedural knowledge (e.g., the BMI formula and classification thresholds). This aligns with prior studies on RAG, which emphasise its role in reducing hallucinations and improving factual grounding in generative outputs. Importantly, RAG allowed the model to successfully apply transformations not directly encoded in the entity schema, addressing one of the core research challenges in DMN automation. Another key takeaway is the role of validation in structured generation tasks. Even when outputs appear semantically correct, subtle structural errors (e.g., overlapping rules, non-exclusive branches) can cause runtime failures in DMN engines. The experiment shows that without automatic validation, assessing correctness solely by manual inspection or LLM confidence is insufficient.

Overall, the experiment provides empirical support for a hybrid architecture combining LLMs with schema-aware prompting, retrieval mechanisms, and automatic validation. This architecture not only increases reliability but also paves the way for more complex, explainable, and verifiable LLM-driven decision automation pipelines. Future work should focus on generalising this approach to more diverse domains, formalising retrieval sources, and exploring integration with constraint-based validation tools. To assess whether the observed difference in accuracy between the LLM-only and LLM & RAG configurations was statistically significant, we applied a two-proportion Z-test. This test evaluates whether two independent proportions differ significantly, under the null hypothesis that they are equal.

We define the proportions of successful outputs as follows:

- LLM only. 86 successes out of 100 trials ($p_1 = 0.86$)
- LLM & RAG. 99 successes out of 100 trials ($p_2 = 0.99$)

The Z-statistic for comparing two proportions is calculated. Substituting the values: $z = -3.49$. The corresponding p-value is approximately 0.00048, which is well below the standard alpha threshold of 0.05. The result is statistically significant with $p < 0.001$, providing strong evidence that the improvement in performance observed with RAG is not due to chance. This supports the conclusion that retrieval-augmented generation contributes meaningfully to the accuracy and reliability of DMN table generation.

Proposed Framework

To address the complexity and manual effort involved in constructing DMN decision tables from natural language, we propose a modular framework that leverages the reasoning capabilities of LLMs enhanced with structured schema injection and domain-specific retrieval. The framework is designed to automatically convert textual descriptions of business rules into executable, validated DMN decision models. At its core, the system integrates five key components: a Schema Injector that provides contextual data awareness, a Retrieval Module that supplements missing background knowledge, a Prompt Composer that structures the full input for the LLM, a Generation Module responsible for producing DMN code, and a Validator that ensures the structural and logical correctness of the resulting model. Together, these modules form a closed-loop architecture capable of supporting both straightforward and derived decision logic. This design enables schema-consistent DMN generation with minimal human intervention and supports the transformation of natural language decision logic into robust decision automation artefacts.

Schema Injector

The Schema Injector is responsible for extracting and formatting structured information about the domain entity that will be processed by the DMN model. It collects the list of available input attributes and their types directly from the entity definition and prepares this information in an XML-based format compatible with the DMN context structure. This component does not perform any logical transformations or reasoning. Its sole purpose is to ensure that the language model receives an explicit and complete description of the input data structure. This includes only directly defined properties without interpretation or derivation. By doing so, the Schema Injector guarantees that the LLM has the necessary raw context to generate a DMN table that aligns with the actual structure of the input data.

Retrieval Module (RAG)

The Retrieval Module enhances the generation process by providing the language model with access to external domain knowledge that may be required for accurate DMN construction. This includes background rules, classification thresholds, standard formulas, regulatory constraints, or any predefined business logic that is not explicitly stated in the user's prompt or the entity schema.

The module operates as part of a RAG pipeline, querying a curated collection of domain documents or rule repositories. Retrieved content is selected based on relevance to the task and appended to the model's input prompt. This augmentation enables the LLM to ground its outputs in verified knowledge rather than relying solely on its internal representation. By enriching the prompt with retrieved information, the Retrieval Module reduces hallucinations, improves coverage of edge cases, and supports consistent reproduction of formal decision logic that aligns with real-world standards or organisational policies.

Prompt Composer

The Prompt Composer is responsible for assembling all necessary input elements into a structured prompt that is submitted to the language model. It combines the user's natural language description of the decision logic, the structured entity schema from the Schema Injector, and the external knowledge retrieved by the Retrieval Module into a single coherent input.

In addition to merging these content blocks, the Prompt Composer also appends a set of explicit generation instructions. These include directives that constrain the language model's output to:

- Return only XML content.
- Generate a valid DMN-compliant decision table.
- Strictly follow the DMN XML specification without additional commentary or formatting artifacts.

This combination of structured context and instructional constraints is essential for achieving predictable, machine-executable output. The Prompt Composer ensures that all parts of the prompt are properly formatted and ordered, minimising ambiguity and improving the consistency and quality of the generated DMN code.

LLM-Based Generation Module

The LLM-Based Generation Module is responsible for transforming the structured prompt into a complete DMN decision table using a large language model. This module acts as a one-shot generator, producing the final output in XML format based solely on the prompt provided by upstream components. The model receives a prompt that includes:

- A text description of the decision logic.
- A formalised XML schema of the input entity.
- Relevant external knowledge retrieved from the knowledge base
- A set of strict generation instructions that define output format and constraints.

The prompt instructs the model to generate a DMN decision table that adheres to the official specification, with no additional text, explanation, or formatting outside of the required XML structure. The generated table must define:

- The input and output elements (matching the schema).
- A decision table with complete and mutually exclusive rules,
- Any internal logic necessary to handle derived attributes,
- Proper references to FEEL expressions where applicable.

The generation process is stateless and context-limited: the model has no memory beyond the provided prompt, which places high importance on the completeness and clarity of the input. The model itself does not validate the generated output, meaning that syntactic and logical correctness are not guaranteed without post-processing.

This module is optimised for structured code generation and benefits from prompt engineering techniques such as few-shot examples, output formatting hints, and output delimiters. Its effectiveness depends directly on the combination of schema injection, relevant retrieval, and well-formulated instructions.

DMN Validator

The DMN Validator is responsible for evaluating the structural and logical correctness of the DMN table generated by the language model. It acts as an independent verification layer, ensuring that the output is not only syntactically valid XML but also semantically correct and executable according to the DMN standard. The validation process consists of several stages:

- **XML schema validation.** Confirms that the generated code adheres to the DMN XML specification and can be parsed by standard DMN engines.
- **Structural consistency checks.** Verifies that all referenced inputs, outputs, and decision rules are properly defined and connected.
- **Logical validation.** Applies the generated decision table to a predefined test dataset to confirm that the logic produces correct and complete outputs across all relevant cases.

The validator detects common issues such as missing rules, overlapping conditions, type mismatches, or undeclared input variables. It also supports the use of edge-case test inputs to confirm the robustness and generalizability of the generated logic.

Conclusions

This research presents a novel framework for automating the conversion of natural language descriptions into DMN decision tables, addressing the complexity and manual effort traditionally involved in constructing these models. By leveraging the reasoning capabilities of Large Language Models (LLMs) in conjunction with structured schema injection and domain-specific retrieval, we have developed a system capable of generating accurate, validated DMN decision models with minimal human intervention. At the core of this framework is a modular architecture that integrates five key components: a Schema Injector, Retrieval Module, Prompt Composer, Generation Module, and Validator. This system facilitates schema-consistent DMN generation, supporting both straightforward and more complex decision logic. The hybrid approach combining LLMs with schema-aware prompting, retrieval mechanisms, and automatic validation not only enhances the reliability and accuracy of decision model generation but also provides a pathway for more explainable and verifiable decision automation pipelines. Empirical evidence gathered from our experiments indicates that the integration of Retrieval-Augmented Generation (RAG) significantly improves the accuracy of DMN table generation. A statistical analysis using a two-proportion Z-test revealed that the performance improvements observed in the LLM & RAG configuration were statistically significant ($p < 0.001$), further supporting the conclusion that RAG contributes meaningfully to the accuracy and reliability of the process. In addressing the research questions, the study provides clear and evidence-based answers. First, the integration of RAG demonstrably enhances the accuracy and completeness of LLM-generated DMN decision tables, confirming its effectiveness in grounding model outputs in domain-specific knowledge. Second, the inclusion of schema injection and retrieval mechanisms proved instrumental in reducing structural and logical inconsistencies, ensuring that generated models adhere more closely to the DMN specification while minimising hallucinations and redundancy. Third, the proposed framework shows strong potential to support scalable, explainable, and verifiable decision automation, offering a robust foundation for integrating AI-driven reasoning into practical business decision-making environments.

These findings suggest that the proposed framework represents a promising step toward practical, explainable, and verifiable decision automation. It can serve as a valuable tool for organisations seeking to enhance the transparency, accuracy, and consistency of their decision-making processes. Building on the insights derived from this study, future research should extend the framework to diverse business domains, formalise retrieval sources for domain-specific reasoning, and integrate constraint-based validation mechanisms to further strengthen reliability and scalability in automated decision modelling.

References

- Arco, L., Nápoles, G., Vanhoenshoven, F., Lara, A. L., Casas, G., & Vanhoof, K. (2021). Natural language techniques supporting decision modelers. *Data Mining and Knowledge Discovery*, 35(1), 290–320. <https://doi.org/10.1007/s10618-020-00718-4>
- Berkovitch, Y., Glickman, O., Somech, A., & Wolfson, T. (2025). Generating Tables from the Parametric Knowledge of Language Models. Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing Albuquerque, New Mexico, USA <https://doi.org/10.18653/v1/2025.knowledgenlp-1.4>
- Boonmepit, B., & Suwannasart, T. (2019). Test Case Generation from BPMN with DMN. Proceedings of the 2019 3rd International Conference on Software and E-Business, 92-96. <https://doi.org/10.1145/3374549.3374582>
- Bork, D., Ali, S. J., & Dinev, G. M. (2023). AI-Enhanced Hybrid Decision Management. *Business & Information Systems Engineering*, 65(2), 179-199. <https://doi.org/10.1007/s12599-023-00790-2>
- Calvanese, D., Dumas, M., Laurson, Ü., Maggi, F. M., Montali, M., & Teinmaa, I. (2018). Semantics, Analysis and Simplification of DMN Decision Tables. *Information Systems*, 78, 112-125. <https://doi.org/10.1016/j.is.2018.01.010>
- Calvanese, D., Dumas, M., Maggi, F. M., & Montali, M. (2018). Semantic DMN: Formalising and Reasoning About Decisions in the Presence of Background Knowledge. <https://doi.org/10.1017/S1471068418000479>
- Corea, C., Blatt, J., & Delfmann, P. (2019). A tool for decision logic verification in DMN decision tables. CEUR Workshop Proceedings, 2420. <https://ceur-ws.org/Vol-2420/papeDT11.pdf>
- Corea, C., Kampik, T., Montali, M. (2024). Explainable DMN. In: Marrella, A., Resinas, M., Jans, M., Rosemann, M. (eds) Business Process Management Forum. BPM 2024. Lecture Notes in Business Information Processing, vol 526. Springer, Cham. https://doi.org/10.1007/978-3-031-70418-5_4
- Estrada-Torres, B., del-Río-Ortega, A., & Resinas, M. (2024). Automatically generating decision-support chatbots based on DMN models. <https://doi.org/10.48550/arXiv.2405.09645>
- Etikala, V. (2021). Extracting decision model components from natural language text for automated business decision modelling. CEUR Workshop Proceedings, 2956. <https://ceur-ws.org/Vol-2956/paper51.pdf>
- Etinger, D., Simić, S. D., & Buljubašić, L. (2019). Automated decision-making with DMN: from decision trees to decision tables. 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 1309-1313. <https://doi.org/10.23919/MIPRO.2019.8756694>
- Fernandez, R. C., Elmore, A. J., Franklin, M. J., Krishnan, S., & Tan, C. (2023). How Large Language Models Will Disrupt Data Management. Proceedings of the VLDB Endowment, 16(11), 3302-3309. <https://doi.org/10.14778/3611479.3611527>
- Funk, D. (2023). Creating a Low-Code Business Process Execution Platform With Python, BPMN, and DMN. *IEEE Software*, 40(1), 9–17. <https://doi.org/10.1109/MS.2022.3212033>
- Gavaza, B. K. (2025). Drivers of Artificial Intelligence adoption in township small businesses in South Africa. *Insights into Regional Development*, 7(4), 116-129. <https://doi.org/10.70132/c9883956762>
- Goossens, A., de Smedt, J., & Vanthienen, J. (2023). Extracting Decision Model and Notation models from text using deep learning techniques. *Expert Systems with Applications*, 211, 118667. <https://doi.org/10.1016/j.eswa.2022.118667>
- Goossens, A., de Smedt, J., & Vanthienen, J. (2024). Comparing the Performance of GPT-3 with BERT for Decision Requirements Modeling. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 14353 LNCS. https://doi.org/10.1007/978-3-031-46846-9_26
- Jeong, C. (2023). A Study on the Implementation of Generative AI Services Using an Enterprise Data-Based LLM Application Architecture. *Advances in Artificial Intelligence and Machine Learning*, 03(04), 1588-1618. <https://doi.org/10.54364/AAIML.2023.1191>
- Kanyuka, A., & Mahfoud, E. (2024). Domain-Specific Shorthand for Generation Based on Context-Free Grammar. <https://doi.org/10.48550/arXiv.2406.10442>
- Leewis, S., Smit, K., & Versendaal, J. (2024). Discovering operational decisions from data—a framework supporting decision discovery from data. *Decision*, 51(4), 417-436. <https://doi.org/10.1007/s40622-024-00402-2>
- Li, H., Su, Y., Cai, D., Wang, Y., & Liu, L. (2022). A Survey on Retrieval-Augmented Text Generation. <https://doi.org/10.48550/arXiv.2202.01110>

Liu, J., Lin, J., & Liu, Y. (2024). How Much Can RAG Help the Reasoning of LLM? <https://doi.org/10.48550/arXiv.2410.02338>

Mugunzva, F. I., & Manchidi, N. H. (2024). Re-envisioning the artificial intelligence-entrepreneurship nexus: a pioneering synthesis and future pathways. *Insights into Regional Development*, 6(3), 71-84. <https://doi.org/10.70132/k5546584395>

Salinas, A., & Morstatter, F. (2024). The Butterfly Effect of Altering Prompts: How Small Changes and Jailbreaks Affect Large Language Model Performance. Findings of the Association for Computational Linguistics ACL 2024, 4629–4651. <https://doi.org/10.18653/v1/2024.findings-acl.275>

Shorten, C., Pierse, C., Smith, T. B., Cardenas, E., Sharma, A., Trengrove, J., & van Luijt, B. (2024). StructuredRAG: JSON Response Formatting with Large Language Models.

Španková, J. & Sochuľáková, J. (2025). The use of artificial intelligence in human resource management of enterprises, *Entrepreneurship and Sustainability Issues*, 13(2), 238-249. <https://doi.org/10.9770/q2447759355>

Tam, Z. R., Wu, C.-K., Tsai, Y.-L., Lin, C.-Y., Lee, H., & Chen, Y.-N. (2024). Let Me Speak Freely? A Study On The Impact Of Format Restrictions On Large Language Model Performance. Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, 1218–1236. <https://doi.org/10.18653/v1/2024.emnlp-industry.91>

Tavantzis, T., Promikyridis, R., & Tambouris, E. (2023). Towards exploiting BPMN and DMN in public service modeling. Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics, 211–216. <https://doi.org/10.1145/3635059.3635092>

van Woensel, W., & Motie, S. (2024). NLP4PBM: A Systematic Review on Process Extraction using Natural Language Processing with Rule-based. *Machine and Deep Learning Methods*, Article: 2417404. <https://doi.org/10.1080/17517575.2024.2417404>

Vandeveldel, S., Aerts, B., & Vennekens, J. (2023). Tackling the DM Challenges with cDMN: A Tight Integration of DMN and Constraint Reasoning. *Theory and Practice of Logic Programming*, 23(3), 535–558. <https://doi.org/10.1017/S1471068421000491>

Funding: This research was funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I03-03-V01-00078

Author Contributions: Conceptualisation: *Cherednichenko O., Maliarenko V.*; methodology: *Cherednichenko O., Maliarenko V.*; data analysis: *Cherednichenko O., Maliarenko V.*, writing—original draft preparation: *Maliarenko V.*, writing; review and editing: *Cherednichenko O., Tancošová J.*; visualisation: *Maliarenko V., Nebeský L.*. All authors have read and agreed to the published version of the manuscript.

Dr. Olga CHEREDNICHENKO, DSc is researcher at the Department of Small and Medium Enterprises of Bratislava University of Economics and Management (Slovakia). Research interests: the mathematical models of intelligent systems, natural language processing, business intelligence.

ORCID ID: <https://orcid.org/0000-0002-9391-5220>

Vladyslav MALIARENKO is PhD student at the Department of Project Management in Information Technology of National Technical University “Kharkiv Polytechnic Institute” (Ukraine). Research interests: decision making, structured data generating, artificial intelligence.

ORCID ID: <https://orcid.org/0009-0009-6064-061X>

prof. Ing. Judita TÁNCOŠOVÁ, CSc. is a professor at the Department of Economics and Finance of Bratislava University of Economics and Management (Slovakia). Research interests: international business and management, microeconomic and macroeconomic problems.

ORCID ID: <https://orcid.org/0000-0001-8963-6280>

PhDr. Mgr. Ľubomír NEBESKÝ, PhD. is an assistant professor at the Department of Economics and Finance of Bratislava University of Economics and Management (Slovakia). Research interests: international business and management, decision making.

ORCID ID: <https://orcid.org/0000-0002-2148-7212>

Copyright © 2026 by author(s). Publishing rights by Entrepreneurship and Sustainability Center

This work is licensed under the Creative Commons Attribution International License (CC BY).

<http://creativecommons.org/licenses/by/4.0/>



Open Access