UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Software Engineering Curriculum

Chioma Nkem-Eze

# Design and Evaluation of an AI-Assisted COMPS Tutor for Students with Learning Difficulties in Mathematics

Master's Thesis (30 ECTS)

Supervisors:
Eduard Barbu, PhD
Kateryna Lipmaa, PhD

Tartu 2025

# Design and Evaluation of an AI-Assisted COMPS Tutor for Students with Learning Difficulties in Mathematics

**Abstract:**
This thesis presents Nutikas, an AI-assisted tutor that automates Conceptual Model-Based Problem Solving (COMPS) for early-grade additive word problems, designed with learners with special educational needs (SEN) in mind. Nutikas uses a four-step prompt pipeline: (i) super-category classification (Change / Combine / Compare), (ii) 12-way subtype selection, (iii) schema slot filling (e.g., Start/Change/End), and (iv) story-grammar questions to align large language model (LLM) outputs with instructional scaffolds. Three current LLMs (GPT-4.1, Claude Sonnet 4, Gemini 2.5 Flash) are evaluated on a 120-item corpus covering all COMPS additive subtypes and score four dimensions: category, subtype, mapping (equation fidelity), and answer. Answers are near ceiling ($\geq$99.2%), while residual errors concentrate in schema mapping especially the polarity of Change-Separate problems where the COMPS convention requires a non-negative change magnitude. Mapping accuracy ranges Gemini 98.3%, Claude 91.7%, GPT-4.1 85.0%, suggesting that remaining variance reflects representation conventions rather than arithmetic capability. A small usability pilot with two SEN students (SUS-Kids mean 68.8) and one teacher indicates acceptable usability and highlights the need for clearer analytics on the teacher dashboard. While Tier-2 findings are formative and the scope is additive only, Nutikas already delivers accurate solutions with actionable paths to close the remaining mapping gap.

**Keywords:**
special education, COMPS, word problems, large language models, schema mapping, story-grammar, usability.

**CERCS:** P170 Computer science, numerical analysis, systems, control

## Tehisintellektiga abistatava COMPS-juhendaja disain ja hindamine matemaatika õpiraskustega õpilastele

**Lühikokkuvõte:**

See lõputöö tutvustab Nutikast, tehisintellektil põhinevat juhendajat, mis automatiseerib kontseptuaalse mudelipõhist probleemide lahendamist (COMPS) algklasside aditiivsete tekstiülesannete jaoks, pidades silmas erivajadustega õppijaid. Nutikas kasutab neljaastmelist ülesannete lahendamise protsessi: (i) ülemkategooriate klassifitseerimine (Muuda / Kombineeri / Võrdle), (ii) 12-suunaline alatüübi valik, (iii) skeemipesade täitmine (nt Algus/Muuda/Lõpp) ja (iv) jutu-grammatika küsimused, et viia suurte keelemudelite (LLM) väljundid vastavusse õppestruktuuridega. Hindame kolme praegust LLM-i (GPT-4.1, Claude Sonnet 4, Gemini 2.5 Flash) 120-punktilisel korpusel, mis hõlmab kõiki COMPS-i aditiivseid alatüüpe, ja hindame nelja dimensiooni: kategooria, alatüüp, kaardistamine (võrrandi täpsus) ja vastus. Vastused on ülemmäära lähedal ($\geq$99,2%), samas kui jääkvead koonduvad skeemikaardistamisele, eriti muutmis-eraldamisülesannete polaarsusele, kus COMPS-i konventsioon nõuab mittenegatiivset muutuse suurusjärku. Kaardistamise täpsus jääb vahemikku Gemini 98,3%, Claude 91,7% ja GPT-4.1 85,0%, mis viitab sellele, et järelejäänud dispersioon peegeldab pigem esituskonventsioone kui aritmeetilist võimekust. Väike kasutatavuse pilootprojekt kahe SEN-õpilase (SUS-Kids keskmine 68,8) ja ühe õpetajaga näitab vastuvõetavat kasutatavust ja rõhutab vajadust selgema analüütika järele õpetaja armatuurlaual. Kuigi 2. taseme tulemused on formatiivsed ja ulatus ainult aditiivne, pakub Nutikas juba täpseid lahendusi koos tegutsemisvõimalustega ülejäänud kaardistamislünga täitmiseks.

**Võtmesõnad:**
eripedagoogika, COMPS, tekstiülesanded, suured keelemudelid, skeemide kaardistamine, loo grammatika, kasutatavus.

**CERCS:** P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

# Contents

# List of Figures

# List of Tables

# 1 Introduction

For many students, solving mathematical word problems is like solving a riddle in two languages at once. They must decode a short story and translate it into mathematics while holding key information in memory. For students with special educational needs (SEN), that is, learners who require additional support to access the mathematics curriculum, including but not limited to those with formally diagnosed learning disabilities, this challenge is magnified. For example, on the 2019 National Assessment of Educational Progress (NAEP) mathematics assessment, about 40% of U.S. fourth graders scored at or above NAEP Proficient [1]. Many questions on large-scale assessments require reading and interpreting problem narratives, which effectively makes word-problem competence a gatekeeper for measured performance [2]. These difficulties with word problems often persist into later grades, restricting access to advanced mathematics and to everyday quantitative problem solving. In Estonia, administrative statistics indicate a high prevalence of students with SEN. In 2014 there were almost 26,000 learners with SEN, and the share of learners with SEN had risen from 13.9% (2006) to 17.1–18.5% (2014), suggesting that roughly one in five students requires additional support [3].

A substantial body of research converges on the same core issue: many students with SEN struggle less with calculation than with modelling the quantitative relationships in a problem's story [4]. Learners may focus on isolated numbers or "trigger words" (e.g., *more*, *altogether*), applying operations by guesswork rather than representing the underlying structure [5]. One method shown to disrupt this cycle is *COMPS* [6]. COMPS teaches students to represent a situation with a small set of generalized model equations and to map story details into those roles via story-grammar prompts [7]. These scaffolds reduce cognitive load and support flexible reasoning across unknown positions [6]. Studies report improved word-problem performance and representation quality for struggling learners [8, 9].

At the same time, advances in large language models (LLMs) (e.g., GPT-4) have made it feasible for artificial intelligence (AI) systems to parse natural language, extract quantities, and generate step-by-step solution plans for word problems [10, 11, 12, 13]. Academic prototypes [12, 13] and commercial apps (e.g., Photomath; QANDA) [14, 15] now produce worked solutions that appear instructional. However, independent assessments suggest these explanations are predominantly procedural rather than conceptually organized, and there was no public documentation found of widely used AI solvers that implement COMPS's generalized model equations and story-grammar prompts.

This thesis addresses that gap. It proposes **Nutikas** (Estonian for "smart"), an AI-assisted tutor that aims to incorporate COMPS principles into word problem instruction. It seeks to replicate the way expert teachers use COMPS while adding the scalability and adaptability of modern AI.

## 1.1 Research Goals

The primary goal is to investigate whether an AI system can provide COMPS-aligned support for early-grade additive word problems in a reliable and instructionally useful way.

**O1: Design and prototype Nutikas** to solve additive word problems (Part–Part–Whole (PPW), Additive Compare (AC)) and to generate COMPS-aligned equations and story-grammar prompts.

**O2: Evaluate accuracy and representation fidelity** against expert annotations (type classification, model–equation fidelity including unknown positions, and answer correctness).

**O3: Assess usability and instructional potential** through student observations and teacher feedback.

## 1.2 Research Questions

**RQ1: COMPS classification accuracy.** How accurately does *Nutikas* classify additive word problems into the COMPS-defined families and subtypes (e.g., PPW vs. AC)?

**RQ2: Equation fidelity and answer accuracy.** How faithful are *Nutikas*'s COMPS-aligned equations, including unknown positions to expert annotations, and how accurate are the resulting answers?

**RQ3: Usability and instructional value.** What is the perceived usability of *Nutikas* for students and the perceived instructional value for a teacher?

## 1.3 Significance of the Study

By integrating an evidence-based teaching framework (COMPS) with the generative capabilities of contemporary AI, this project aims to move from "AI that answers" toward "AI that teaches". If effective, **Nutikas** could provide high-quality, conceptual problem-solving practice at scale for SEN learners and a practical differentiation aid for teachers. In the context of Estonia's ongoing investment in digital learning [16], this work illustrates how principled pedagogy and state-of-the-art AI may contribute to fairness and inclusion in mathematics education.

# 2 Background and Related Work

This chapter reviews research that guides the design and evaluation of Nutikas, an AI-assisted tutoring system for students with SEN solving additive word problems. It first outlines COMPS, the instructional framework underpinning Nutikas, summarizing its pedagogy and evidence of effectiveness for SEN learners. It then surveys technology-based interventions for word-problem instruction in special education, highlighting their strengths and current limitations. The chapter concludes with recent advances in LLMs, emphasizing their potential to automate COMPS-aligned strategies and mitigate scalability challenges in personalized, research-driven mathematics instruction.

## 2.1 Conceptual Model-Based Problem Solving (COMPS)

COMPS is a specialized form of model-based problem solving related to, but distinct from, schema-based instruction (SBI). Whereas SBI teaches students to classify problems into schemas and apply fixed rules or diagrams for each type [9], COMPS uses generalized model equations that capture relationships common to multiple problem types, reducing memory load and supporting transfer to new contexts [6]. A key design element is the use of story-grammar prompts, guided comprehension questions that help students map from narrative text to a quantitative model [7]. Typical prompts include: Who/what is involved? What are the parts? What is the whole? How are the quantities related?

**Example.** Consider:

> *There are 8 red apples and 5 green apples in a basket. How many apples are there altogether?*

Story-grammar prompt mapping: Who/what: red apples, green apples; Parts: 8 and 5; Whole: unknown; Relation: Part + Part = Whole. The corresponding model equation is $8 + 5 = W$.

A distinctive advantage of COMPS is that the same generalized equation accommodates various unknown positions. In the following sections, this study clarifies the unknown positions for the additive families employed.

### 2.1.1 Additive Problem Families and Unknown Positions

COMPS organizes problems into three families [6]: **Part-Part-Whole (PPW)**, **Additive Compare (AC)**, and **Equal Groups (EG)**. This thesis focuses on the *additive* families (PPW and AC) and excludes EG to keep scope manageable and because PPW/AC is core in early curricula and remain challenging for many students with SEN [17]. For each family, the generalized model equation admits three unknown positions.

**PPW (Combine).**

$$\begin{aligned} \text{Whole} &= \text{Part}_1 + \text{Part}_2, \\ \text{(whole unknown)} \quad W &= p_1 + p_2, \\ \text{(part unknown)} \quad p_1 &= W - p_2, \quad p_2 = W - p_1. \end{aligned}$$

**PPW (Change).**

$$\begin{aligned} \text{End} &= \text{Start} \pm \text{Change}, \\ \text{(end unknown)} \quad E &= S \pm C, \\ \text{(start unknown)} \quad S &= E \mp C, \quad \text{(change unknown)} \quad C = |E - S|. \end{aligned}$$

*Note:* $C$ is a non-negative magnitude by convention in COMPS.

**AC (Compare).**

$$\begin{aligned} \text{Bigger} &= \text{Smaller} + \text{Difference}, \\ \text{(difference unknown)} \quad D &= B - S, \\ \text{(bigger unknown)} \quad B &= S + D, \quad \text{(smaller unknown)} \quad S = B - D. \end{aligned}$$

Table 1 summarizes the additive structures considered in this thesis, including representative unknown positions.

| Problem Type | Generalized Model | Examples with Unknown Positions |
|---|---|---|
| PPW; Combine | $W = p_1 + p_2$ | Whole unk.: "8 red and 5 green apples. How many altogether?" $\Rightarrow 8 + 5 = W$. Part unk.: "There are 13 apples; 5 are green. How many red?" $\Rightarrow p = 13 - 5$. |
| PPW; Change | $E = S \pm C$ | End unk. (increase): "Sarah had 9 candies and got 4 more. How many now?" $\Rightarrow 9 + 4 = E$. End unk. (decrease): "Leo had 12 stickers and gave away 5. How many now?" $\Rightarrow 12 - 5 = E$. |
| AC; Compare | $B = S + D$ | Difference unk.: "Tom has 12, Sam 7. How many more does Tom have?" $\Rightarrow D = 12 - 7$. Smaller unk.: "Tom has 12, which is 5 more than Sam. How many does Sam have?" $\Rightarrow S = 12 - 5$. |

**Table 1.** Additive COMPS types with models and example unknown positions.

### 2.1.2 Why COMPS for learners with SEN?

Multiple studies report that COMPS improves how well a student captures the structure of a word problem, word problem solving accuracy, and readiness for algebraic reasoning for students with mathematics learning difficulties [7, 6, 8]. Its design aligns with evidence-based principles for SEN mathematics instruction [17]:

- **Consistency.** A compact set of generalized model equations applies across contexts, lowering memory demands.
- **Multiple representations.** Visual diagrams linked to symbolic equations support conceptual understanding.
- **Guided comprehension.** story-grammar prompt prompts scaffold mapping from narrative to model [7].

Meta-analyses of word problems intervention show that explicit instruction, visual supports, and guided practice; features central to COMPS, yield significant learning gains for students with learning difficulties [9, 18]. However, these reviews encompass a

range of models, so COMPS-specific results should be interpreted within that broader evidence base.

## 2.2 Technology for Word-Problem Solving in Special Education

Over the past two decades, researchers have built a range of technology-based tools to support students with SEN in solving arithmetic word problems. Early computer-assisted instruction (CAI) provided stepwise instruction, targeted practice, and immediate feedback [19]. Many systems used visual cueing (e.g., color highlighting) to direct attention during problem reading. In one eye-tracking study, the clearest gains were on attentional measures rather than accuracy per se [20]. Other designs integrated structured supports (e.g., graphic organizers, diagrams) to help identify problem type, extract essential information, and represent it in solvable form, useful for learners with working-memory or executive-function needs. Multimedia environments combined text, audio, and animation to model solution steps; virtual manipulatives helped connect abstract quantities to concrete representations, particularly in elementary grades [19].

Within this landscape, some systems adopt COMPS (see §2.1) explicitly. For example, a multiplicative COMPS tutor (often referred to as PGBM-COMPS) reported improvements on both researcher-developed and standardized measures [21]. A web-based COMPS tutor for additive problems combined interactive diagrams, generalized model equations, and guided story-grammar prompts, improving performance for struggling learners [8, 22].

**Strengths.** Controlled studies suggest that technology-based interventions can substantially improve word-problem performance for SEN learners, especially when explicit instruction, visual representations, and immediate feedback are combined [19]. COMPS implementations, in particular, are linked to better problem classification, more accurate equation formulation, and transfer to new problem types [21, 8, 22]. Digital delivery also supports consistent presentation of instructional steps, extensive practice, and multimodal supports that can enhance accessibility.

**Limitations.** Many SEN-focused word-problem tools, including COMPS systems, have been evaluated in relatively small research settings and often rely on close adult facilitation, which can limit scalability to typical classrooms [19, 21, 22]. Adaptivity is frequently shallow: feedback is immediate but not consistently tailored to specific misconceptions, reflecting CAI-style tutorials more than full intelligent tutoring systems (ITS) with a detailed learner model [19, 23].

## 2.3 Large Language Models for Word-Problem Solving

Recent LLMs (e.g., GPT–4) exhibit strong natural-language understanding and emergent reasoning skills [10], making them attractive for education. They can interpret diverse problem phrasings, identify underlying mathematical structure, and generate worked solutions in natural language. Unlike fixed-script CAI, LLMs can generate novel problems, adapt difficulty, and evaluate student responses, explaining errors in accessible language when prompted appropriately. These properties motivate using LLMs to build more adaptive and scalable supports for word problems [24].

**Current limitations.** Out-of-the-box LLM solvers tend to emphasize procedural narratives and may omit the relational structure that COMPS teaches, which is critical for SEN learners [25]. They can also produce plausible but incorrect outputs (hallucinations). Commercial tools (e.g., Photomath; QANDA) demonstrate real-time solving but rarely include graduated visual scaffolds, guided story-grammar prompts, or misconception-specific feedback; their design is typically oriented toward independent study rather than structured pedagogy [14, 15]. Moreover, there is limited research on LLM effectiveness with SEN populations, so instructional impact remains largely untested.

## 2.4 Summary and Research Gap

Three strands of prior work shape this study. First, COMPS has demonstrated effectiveness in helping students with mathematics learning difficulties represent and solve word problems by making problem narative structure explicit [6, 7, 8]. Second, technology-based implementations, including COMPS tutors improve accuracy, transfer, and engagement when they combine visual representations, step-by-step guidance, and immediate feedback [19, 21]. However, many systems require substantial teacher facilitation and offer only modest adaptivity [23]. Third, recent LLMs offer a path to overcome some of these limitations via scalable, natural-language interaction [10, 24]. Yet most LLM-based solvers lack COMPS-style conceptual scaffolding and SEN-specific supports, and their instructional effectiveness in SEN contexts is not yet established [25].

**Gap addressed.** To our knowledge, there is no evaluated system that explicitly implements COMPS (story-grammar + generalized equations) using an LLM toolchain for SEN learners. This thesis addresses that gap by designing and evaluating an AI-powered learning environment that uses LLMs to deliver COMPS-style instruction adaptively, aiming to combine the scalability of modern AI with the evidence-based pedagogy required for effective SEN support.

# 3 Methodology

This chapter details the design, development, and evaluation of *Nutikas*, an AI-assisted platform for arithmetic word problems built on the COMPS framework. The methodology combines an LLM prompt pipeline, a web application, and a two-tier evaluation that links model behaviour to learning-oriented outcomes.

1. **Prompt pipeline** (§3.1): automates COMPS via problem classification, model-equation scaffolding, and story-grammar prompt generation.
2. **Web application** (§3.2): student quiz interface, teacher dashboard, and the inference service that supplies COMPS-aligned scaffolds.
3. **Evaluation** (§3.3): *Tier 1* scores LLMs against expert ground truth (type classification and mapping/equation fidelity); *Tier 2* examines usability and perceived instructional value.

Together, these components address the study's research questions (RQ RQ1–RQ3). The sections that follow present, in order, (i) the prompt pipeline and its design rationale, (ii) the system architecture, and (iii) the evaluation procedures and rubrics. This mirrors the overall flow of the thesis: prompts shape the outputs, the application delivers them, and the evaluation interprets them.

## 3.1 Prompt-Design Overview

**Prompting as a specification layer.** Prompting was used as the specification layer that turns problem narratives into structured COMPS representations, COMPS-aligned equations, and brief story-grammar prompts. In this study, prompts constrained both the *form* (labels, slots, JSON fields) and the *language* (reading level, slot phrasing) so that outputs are consistent, scorable, and machine-parsable.

A four-prompt pipeline was implemented and executed identically across three LLMs (GPT-4.1, Claude Sonnet 4, Gemini 2.5 Flash). The pipeline covered: (i) COMPS family and subtype classification, (ii) slot mapping and model-equation scaffolding (including unknown positions), (iii) generation of age-appropriate story-grammar prompts, and (iv) strict JSON packaging.

This design underpinned the three research questions. For **RQ1**, a fixed COMPS taxonomy and explicit decision rules (e.g., non-negative *change*; multi-step actions are summed) were specified, and a strict JSON schema ensured reliable parsing of the predicted *family* and *subtype* labels for classification accuracy calculations. For **RQ2**, the slot-mapping and equation-scaffold prompts enforced *COMPS-aligned structure* and enabled fidelity comparisons to expert annotations and answer accuracy. For **RQ3**, concise templates and consistent wording were specified to support readability; their

*effect on use* was assessed in Tier 2 via SUS-Kids and think-aloud/teacher feedback. The full prompt texts are reproduced in Appendix .1 and **??**.

### 3.1.1 Super-category classification (Change, Combine, Compare)

The initial prompt (see Appendix .1) asks the model to identify the broad situation type for each word problem so the UI can present the appropriate equation scaffold. Its design draws on two strands:

- **COMPS pedagogy.** Teach the three additive schemas; Change, Combine, Compare, with language that highlights temporal change, aggregation, or contrast so students can align the text to the model [6, 7].
- **Prompt-engineering practices.** role priming, guided chain-of-thought, and strict schema-constrained output (e.g., JavaScript Object Notation (JSON)) improve stylistic consistency, reasoning quality, and parsing reliability [26].

The prompt has four parts:

- **Role priming.** Casts the model as "a math educator trained in COMPS," anchoring responses in instructional language rather than casual chat.
- **Category reference cards.** Three short definition cards for Change, Combine, and Compare give (i) a one-line description and (ii) "common indicators." Cue verbs such as *got*, *lost*, and *joined* make clear the difference between temporal change, aggregation, and contrast, distinctions early pilots showed LLMs often miss.
- **Guided reasoning checklist** (chain-of-thought). A "think step by step" checklist that focuses attention on entities, quantities, and the underlying relation before labeling.
- **Fixed machine-readable output** (JSON). A strict JSON schema with the chosen category and a one-sentence rationale yields parsable output and a clear trace for error analysis.

A recurring pilot error was tagging time-referenced problems as Combine merely because they mention two time points (e.g., "on Monday…, on Tuesday…"). The Change card is refined to stress *before–after transformation* (e.g., *received*, *lost*, *joined*) and the notion of a *single quantity* changing state (not two static parts). This reduced those misclassifications. The next subsections show how the same role framing, reasoning checklist, and output contract are adapted for the remaining prompts in the pipeline.

### 3.1.2 Subtype classification

After labeling a word problem as *Change*, *Combine*, or *Compare*, a second prompt assigns one of the twelve COMPS subtypes (see Appendix .5 in Appendix .5). The prompt loads the rules for the chosen super-category, verifies the situation pattern, and then identifies the missing quantity. The full template is in Appendices .1, .3

**Change.** The model first decides whether the quantity *increases* (Join) or *decreases* (Separate) over time. It then maps story numbers to the Change schema (`start`, `change`, `end`), marking any absent value as `unknown`. Given the labeled slots, the subtype is determined by the *role* of the unknown (whole vs. part), not its textual position (see the decision card in Appendix .6.1). This avoids a common pilot error where questions placed at the end of a sentence led to misclassification. For example: "There were some apples. John took 8, his sister took 5, and 7 remain—how many were there initially?" is `CSWU` (unknown *Start*), not `CJWU`.

**Combine.** The model uses the Combine schema (`part1` + `part2` = `whole`), extracts the two parts and the whole from the narrative, and marks the missing slot as `unknown`. The subtype then follows directly from whether a *part* is missing (`CPU`) or the *whole* is missing (`CWU`). See the decision card in Appendix .6.2.

**Compare.** The model analyzes the comparison language (e.g., "how many fewer/less than" vs. "how many more than") to select *Compare–Less* or *Compare–More*. It then fills the Compare schema (`bigger` = `smaller` + `difference`), marking the missing slot as `unknown`. Mapping must follow the original wording (e.g., "A is 5 fewer than B" $\Rightarrow$ `bigger` = B, `smaller` = A, `difference` = 5). The prompt explicitly forbids paraphrasing to prevent flips (fewer $\leftrightarrow$ more). The subtype is chosen solely by the missing role: _DU (difference), _LQU (larger quantity), or _SQU (smaller quantity). See the decision card in Appendix .6.3.

### 3.1.3 Numerical model and answer extraction

Following subtype classification, the third prompt extracts the numerical values for the relevant COMPS equation and computes the missing value. The model returns a single JSON object with: (i) the category-specific slots (e.g., `start`, `change`, `end` for *Change*; `part1`, `part2`, `whole` for *Combine*; `bigger`, `smaller`, `difference` for *Compare*), (ii) a numeric `answer`, and (iii) a one-sentence `reasoning`. Slot values must be numbers or `null` (no expressions), which allows the UI to render a canonical scaffold and enables automatic checks of mapping fidelity and arithmetic against expert ground truth. See Appendix .3 for the full prompt template.

### 3.1.4 Story-grammar prompt generation

The fourth prompt generates story-grammar questions aligned to the chosen COMPS *subtype*. Given the `subtype` and the filled model slots, the model must return a JSON array only. Each element corresponds to one schema slot and must contain exactly the keys `text`, `boxTarget`, and `context` (all plain strings; no additional fields). See Appendix .4 for the full template and the allowed `boxTarget` values per subtype.

**Guards and rationale.** To ensure faithful, parsable, classroom-ready questions, the following are enforced: (i) *no leakage*; the computed `answer` is provided to the prompt for internal consistency but must not be revealed; outputs are checked for leakage during evaluation; and (ii) *age-appropriate language*; avoid technical terms and prefer concrete nouns from the story. These constraints support the pedagogical goal (mapping text spans to schema slots with simple instructor language) and enable deterministic UI rendering and automated validation.

## 3.2 System Design & Development

Nutikas was implemented as a modular web application comprising a student interface, a teacher dashboard, and an offline AI evaluator. The student interface operationalises COMPS scaffolds and provides the interaction flows used in the pilot. The teacher dashboard organises quiz attempts, mappings, and outcomes into classroom-facing views. The AI evaluator pre-computes and stores model outputs under a strict JSON schema so that family/subtype labels, slot mappings, equations (including unknown positions), and answers can be scored reproducibly.

### 3.2.1 Architecture Overview

**Monorepo layout.** The implementation is organized as a monorepo with three bounded components:

- **Client:** Next.js app for the student quiz interface and teacher dashboard.
- **Server:** Express.js API for auth, class/student/quiz management, validating submissions, and serving pre-evaluated problems with structured feedback.
- **AI Evaluator:** Offline utility (development phase) that batch-evaluates problems with multiple LLMs via OpenRouter, persists schema-conformant outputs, and avoids runtime model calls.

**Data and deployment.** All components share a PostgreSQL database via Prisma ORM. The client is deployed on Vercel; the server runs on Render. Environment configuration is injected per environment (dev/test/prod). A high-level diagram appears in Figure 1.

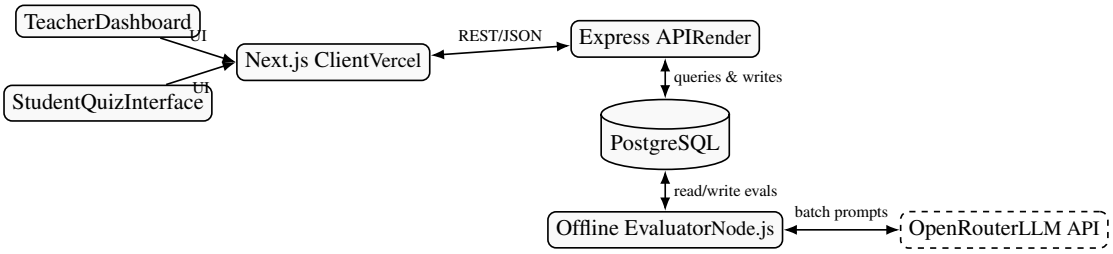



**Figure 1.** High-level architecture: Next.js on Vercel → Express API on Render → PostgreSQL; an offline evaluator uses OpenRouter and writes back results.

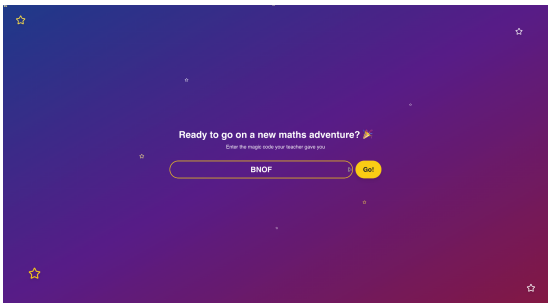
### 3.2.2 Student Interface

The student interface was developed to offer a clear, engaging, and accessible problem-solving experience for early primary grade learners. The visual and interactive components were designed to accommodate students with learning disabilities in mathematics, emphasizing the reduction of cognitive overload and promoting independent exploration. Key features include:

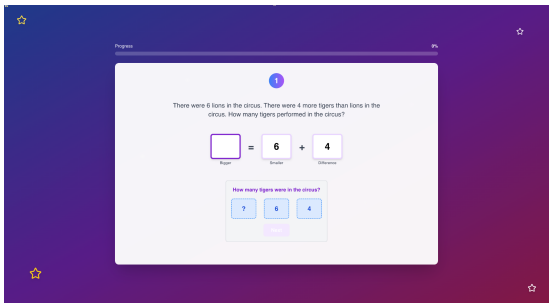
- **Design suitable for children**: The interface incorporates large, legible fonts, a streamlined layout, and high-contrast color schemes. Interactive elements are appropriately spaced and designed for drag-and-drop interaction, thereby minimizing visual clutter and enhancing focus and comprehension.
- **Structured presentation of the problem**: Each session presents a single word problem, positioned at the top of the screen. The problem is articulated in clear, age-appropriate language designed for students in Grades 1 to 4.
- **Interactive model development**: Students are presented with a blank model equation scaffold beneath the word problem (e.g., __ + __ = __). Users engage with this model by dragging numerical values from the problem into designated boxes, facilitating a visual and tactile construction of their comprehension of the problem's structure.
- **Guided story-grammar prompt prompts**: To assist in identifying relevant quantities and relationships, the interface presents a set of AI-generated story-grammar prompt prompts. These appear in a fixed sequence to guide attention and reinforce conceptual understanding (e.g., "What is being added?" or "What is the total?").
- **Answer submission and feedback for retry**: Upon completion of the model, students input their final answer in a designated field and submit their response. The system offers immediate feedback when the answer is incorrect by highlighting the specific area of the model requiring revision, enabling the student to attempt

the question again. Following several unsuccessful attempts, the appropriate model and solution are disclosed for instructional assistance.

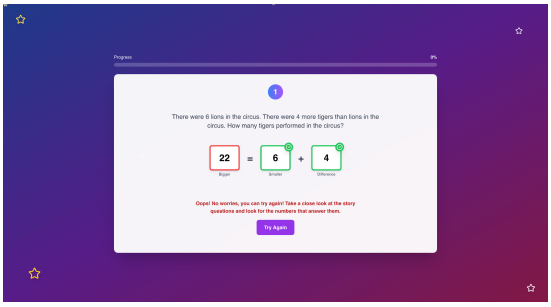
- **Progress monitoring**: A visual progress indicator displays the percentage of problems left to complete the current session. This aids students in maintaining motivation while completing the quiz.



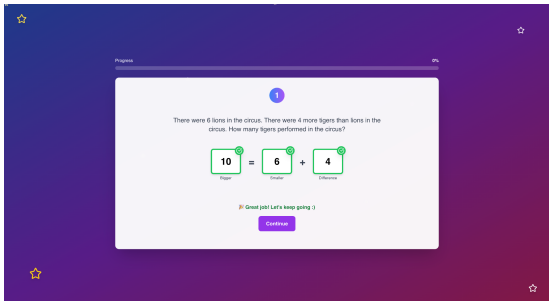

**(a)** Enter session code (no login).




**(b)** Problem with scaffold and first story-grammar prompt.




**(c)** Incorrect attempt; targeted feedback and retry.




**(d)** Correct completion; proceed to next item.

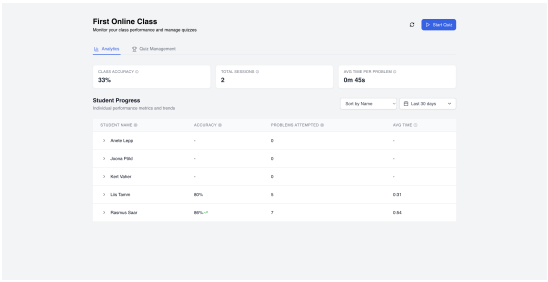
**Figure 2.** Student interaction flow for a *Compare–More* item: (a) access; (b) scaffold + story-grammar; (c) feedback; (d) completion.

### 3.2.3 Teacher Dashboard

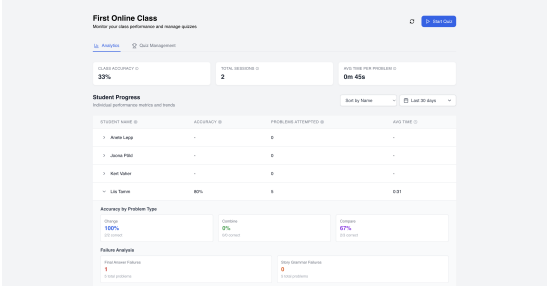
The dashboard supports class setup, creation of quiz sessions, and continuous monitoring, with the goal of making COMPS-aligned practice easy to deploy and act on.

- **Create & assign quizzes.** Build sessions from a pre-classified bank, filterable by COMPS family/subtype and unknown position; each session generates a four-character access code (no logins).
- **Class analytics.** Accuracy, attempts, completion, and time-on-task summaries highlight common misconceptions and outliers.

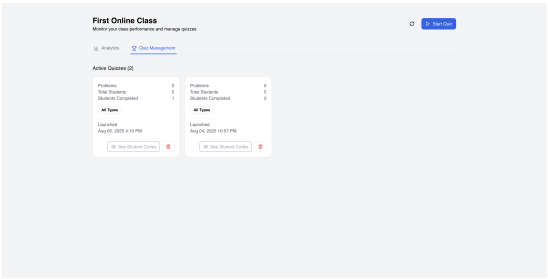
- **Student drill-down.** Per-learner timelines show items attempted, retries, and final status to inform follow-up instruction.



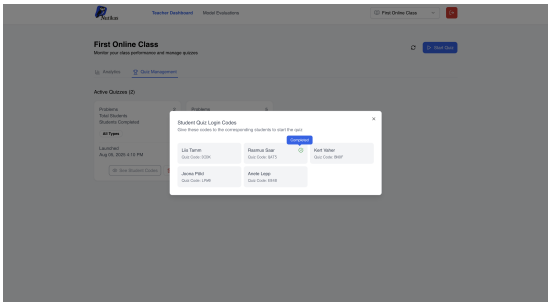

**(a)** Class-wide analytics overview.




**(b)** Individual student analytics.




**(c)** Quiz session management.




**(d)** Student access codes (no logins).

**Figure 3.** Key teacher dashboard views used to run sessions and monitor learning.

### 3.2.4 AI Evaluator

The AI evaluator functions as an offline processing module used to pre-classify and validate word problems prior to their integration into the Nutikas web application. Instead of querying LLMs in real time, which incurs costs, the evaluator was utilized during development to generate and evaluate model outputs across a curated set of arithmetic problems. Key features include:

- **Prompt construction:** For each word problem, the evaluator generates a structured prompt requesting the model to output a classification (e.g., Combine, Compare), subcategory (e.g., Part–Part–Whole), conceptual model equation, story-grammar prompts, step-by-step reasoning, and final numerical answer.
- **Batch model evaluation:** Prompts are sent in batches to selected LLMs via the OpenRouter application programming interface (API). The system accommodates various models (e.g., GPT-4.1, Claude Sonnet 4, Gemini 2.5) while logging metadata such as token usage and response time.

- **Structured output parsing:** Returned outputs are parsed into discrete fields (e.g., model type, story-grammar prompt, final answer) for comparison with ground truths. This facilitates evaluation of mathematical accuracy and adherence to COMPS-aligned representations.
- **Storage and reuse:** All assessed items, together with the generated outputs, are recorded in the database. In live student sessions, only problems that have undergone model validation are presented. This eliminates real-time model calls while maintaining high-quality, AI-generated content.

## 3.3 Evaluation Criteria and Design

The evaluation adopts a two-tier design aligned with the research questions. *Tier 1* quantitatively compares model outputs with expert annotations, covering *COMPS* family/subtype classification, slot-mapping and equation fidelity (including unknown positions), and final answer correctness, addressing RQs RQ1 and RQ2. *Tier 2* is a small-scale, mixed-methods pilot with students and a teacher that examines usability and perceived instructional value (RQ RQ3). The following subsections describe datasets, procedures, and scoring for each tier.

### 3.3.1 Tier 1: AI-Output Performance Evaluation

**Design.** Tier 1 is a quantitative, reproducible evaluation on a fixed, annotated set of 120 additive word problems, balanced across the 12 COMPS subtypes (10 per subtype; drawn from the PPW and AC families). Items were adapted from Estonian special-education materials for Grades 1–3 [16]. Prompt development used a separate pool [27, 28, 29] to avoid overfitting. The frozen dataset, annotations are archived at [30]; the project repository is at [31]. A brief schema appears in Appendix .7.

**Models and procedure.** Three LLMs were evaluated to separate prompt quality from model idiosyncrasies: GPT-4.1 (OpenAI), Claude Sonnet 4 (Anthropic), and Gemini 2.5 Flash (Google DeepMind), selected with reference to OpenRouter's academic rankings [32]. Models were called via OpenRouter with provider defaults; no hyperparameter tuning or programmatic retries. Each item was generated once (single-pass). Outputs had to match a strict JSON contract (category, subtype, slot mapping, answer); non-conforming payloads were scored as errors for the relevant metric. No post-hoc normalization (e.g., sign fixing, recomputing) was applied.

**Metrics.** There are four accuracy components scored which are aligned to COMPS reasoning steps: recognizing the situation, selecting the subtype, mapping numbers to slots, and computing the answer.

| Metric | Definition |
|---|---|
| Super-category accuracy | Correct Change / Combine / Compare label. |
| Sub-category accuracy | Correct 12-way COMPS subtype. |
| Equation fidelity | All numbers placed in the correct schema slots (PPW/AC). |
| Answer correctness | Final numeric answer matches ground truth. |

**Table 2.** Tier 1 accuracy metrics.

**Error categories.** Disaggregating errors localizes failure modes and informs prompt revision.

| Category | Definition |
|---|---|
| Conceptual misclassification | Wrong super-category or subtype. |
| Mapping errors | Correct type but incorrect placement of numerals in schema slots. |
| Computational errors | Correct mapping but incorrect arithmetic outcome. |

**Table 3.** Error categorization used in Tier 1 analysis.

### 3.3.2 Tier 2: End-User Evaluation

**Design.** Given the brief pilot window, a rapid triangulation design was implemented, integrating four small, complementary activities: (1) student think-aloud sessions; (2) an adapted SUS-Kids questionnaire [33]; (3) an exploratory expert usability walkthrough of the app's quiz flow (using the ECA–SEN rubric [34, 35]); and (4) a teacher dashboard review (short rubric plus semi-structured interview). The aim was to examine whether independent strands pointed toward the same conclusions about usability and instructional value. Instruments, full rubrics, and sample materials are provided in Appendices .8, .9, .10, and .11.

**Participants and materials.** Two primary-age students (7–11 years, bilingual, formally identified as having special educational needs) participated in the student-facing activities. Each completed an approximately 30-minute think-aloud session (three to four problems) immediately followed by SUS-Kids [33]. A mathematics–language specialist conducted the expert usability walkthrough of the learner-facing quiz flow. A primary teacher, familiar with SEN learners, reviewed the teacher dashboard using a short rubric and a semi-structured interview guide.

**Sampling and provenance (expert usability walkthrough).** To link Tier 1 content to Tier 2 observations, a 10% simple random sample (12 tasks) was drawn from the 120-item additive COMPS corpus used in Tier 1. During a single quiz session in the deployed *Nutikas* app (Gemini 2.5 Flash back end), the expert worked through all 12 tasks as a learner would. To observe explanatory behaviour, some items were deliberately answered incorrectly to trigger the guidance shown after errors; the remaining items were completed correctly. Judgements focused strictly on *learner-facing* content: clarity of wording, logical flow, scaffolding, and alignment with visual representations. Each task received ECA–SEN ratings.

**Procedures.**

- **Think-aloud (students).** Pupils solved 3–4 Nutikas problems while verbalising their thinking. If silence exceeded $\sim$10 seconds, a neutral nudge (e.g., "Keep talking") was used; no hints or strategy prompts were given. Critical incidents were time-stamped and described. Incidents were later grouped into three thematic codes: *Engagement*, *Navigation friction*, and *Reasoning & representation*, each tagged to a COMPS stage (read $\rightarrow$ map $\rightarrow$ compute). The SEN-friendly think-aloud protocol is provided in Appendix .8.

- **Adapted SUS-Kids (students).** Immediately after the session, students completed a 12-item smiley-Likert questionnaire. The first 10 items were converted to a SUS usability score (0–100) using the standard procedure (subtract 1 from positively worded items; subtract each negatively worded score from 5; sum the adjusted values; multiply by 2.5). Items 11–12 were summarised separately as engagement indicators. A worked scoring example is shown in Appendix .9.

- **ECA–SEN rubric (expert; exploratory) [34, 35].** An expert usability walk-through of the quiz flow was conducted on the sampled tasks. Where feasible, both incorrect and correct completion paths were observed to capture variation in feedback screens. After each task, the learner-facing text and guidance were judged on four dimensions (*Logical flow & coherence*, *Linguistic accessibility*, *Cognitive load & scaffolds*, *Multi-modal alignment*) using a 1–5 scale. Brief notes explained any ratings below 4. As only one rater participated, inter-rater reliability was not estimated (see Limitations). The rubric text appears in Appendix .10.

- **Teacher dashboard review.** A primary teacher completed short task scenarios (e.g., create a class, start a quiz, view results), filled a brief rubric (navigation, onboarding clarity, analytics clarity), and joined a semi-structured interview on instructional use and desired features. Rubric ticks were converted to 1–5 subscores and averaged per criterion; interview notes were open-coded, grouped, and named into broader themes. The teacher rubric and interview prompts are in Appendix .11.

# 4 Results

This section reports results across two tiers aligned with the research questions. Tier 1 quantitatively compares three LLMs: Gemini 2.5 Flash, Claude Sonnet 4, and GPT-4.1, on a 120-item corpus spanning all 12 COMPS subtypes, with performance reported on four COMPS-aligned components (category, subtype, equation/map fidelity, and numeric answer) and a macro-average ("Total"). Tier 2 presents an equally weighted, mixed-methods usability evaluation, combining student think-alouds, an expert walkthrough, and teacher feedback to assess language clarity, cognitive load, multimodal alignment, and dashboard usability.

## 4.1 Tier 1: Model Performance (RQ1, RQ2)

All three models achieved near-ceiling overall accuracy ($\geq$95%). Residual error was driven by equation fidelity/mapping rather than classification or computation of final answer, with Gemini strongest on mapping (98.3%), followed by Claude (91.7%) and GPT-4.1 (85.0%). Greater token use did not coincide with higher mapping accuracy (Claude used the most tokens yet mapped worse than Gemini), suggesting remaining errors reflect slot–value conventions rather than generation length.

| Model | Overall | Family (acc) | Subtype (acc) | Answer (acc) | Equation fidelity | Tokens/item |
|---|---|---|---|---|---|---|
| Gemini 2.5 Flash | 99.58% | 100.0% | 100.0% | 100.0% | 98.3% | 623.9 |
| Claude Sonnet 4 | 97.33% | 99.2% | 99.2% | 99.2% | 91.7% | 697.7 |
| GPT-4.1 | 95.63% | 99.2% | 98.3% | 100.0% | 85.0% | 586.5 |

**Table 4.** Performance across 120 word problems. *Family/Subtype* report COMPS classification accuracy (RQRQ1). *Equation fidelity* is the exact-match rate of COMPS-aligned mappings (RQRQ2). *Tokens/item* is the mean (prompt+completion). *Overall* is the macro-average of Family, Subtype, Answer, and Equation fidelity.

**Coverage.** Every problem (120/120) received a correct numerical answer from at least one model; no item was universally missed. For items with at least one model error, at least one other model produced a schema-conformant mapping.

**Subtype overview** (i) **Combine** subtypes were uniformly perfect across all metrics; (ii) **Change–Join** was near ceiling for all models with one minor mapping dip for Gemini; (iii) **Change-Separate** was the principal bottleneck, showing the largest divergence in mapping accuracy across models; and (iv) **Compare** items were near perfect overall with a few model-specific dips.

### 4.1.1 Error analysis

Out of 120 problems, 21 outputs were assessed as incorrect. In nearly all instances, the numerical responses were correct; inaccuracies were primarily found in the schema mapping that associates story quantities with COMPS roles. This analysis examines three recurring failure modes, their origins, and their impact on scores.

1. **Negative Change in Change–Separate Problems** In Change-Separate problems (CSWU/CSPU), the schema represents change as a non-negative magnitude. In cases where narratives are articulated as decreases (e.g., "spent/gave/lost $k$ ... has $n$ left"), models frequently produced a signed delta, represented as `change = -k`. This internal representation of "net change" is reasonable; however, it violates the schema contract, resulting in a mapping error, even if the computed `start` or `end` is accurate.

   > **Example (CSWU).** "Joonas spent 5 euros and has 7 left; how many originally?" Ground truth: `end = 7`, `change = 5`, `start = 12`. GPT-4.1 and Claude output `change = -5` (mapping error). Gemini output `change = 5` (fully correct).

   **Design Implications.** (i) Clarify the contract in the instruction: "`change is a non-negative magnitude; never include a sign.`" (ii) Add a validator that normalizes signed outputs, e.g., `change := |change|`.

2. **Missing field values in multi-step Change (CJ/CSPU).** A small number of Change-Join and Change-Separate (part-unknown) items returned `change : null` despite explicit quantities in the text, typically when the story mentions two actions that must be aggregated (e.g., "used 10 and 3", "added 2 and 2"). The correct schema requires a single net `change`.

   > **Examples.** *CSPU (eggs)*: "Bought 20; used 10 and 3." Expected: `start = 20`, `change = 13`, `end = null`. Observed: `change = null`.
   > *CJWU (pies)*: "There were 4; added 2 and 2." Expected: `start = 4`, `change = 4`, `end = null`. Observed: `change = null`.

   **Design Implications.** (i) In the instruction, require aggregation: If multiple changes are described, aggregate them and assign change to the non-negative total. (ii) If detailed per-step information is beneficial, introduce an auxiliary field `changes: [10, 3]`, while maintaining the requirement for `change` to be mandatory.

3. **Misclassification and non-conforming outputs (case studies)** In addition to signed and aggregation issues, three examples demonstrate how accurate arithmetic can be associated with an erroneous COMPS label or a non-conforming output.

**Case 1: Compare vs. Change (misclassified by GPT-4.1).** *Prompt:* "The teacher put 17 pictures on the wall. Last week there were 10 pictures on the wall. How many more pictures are on the wall now?"

*Ground truth:* Compare—More, Difference Unknown (CMDU); roles: `bigger=17, smaller=10, difference=?`.

Model reasoning (summarized): GPT-4.1 presented the narrative as a transformation from a previous state to a current state ("last week $\rightarrow$ now"), concluded with the phrase "Join (increase)," and identified Change—Join, Part Unknown (CJPU) with the structure `start (10) + change (?) = end (17)`. The computation of $17 - 10 = 7$ was executed accurately.

Contributing cue conflict: Lexical cues such as *last week*, and *now* activate a Change template, whereas the query *"how many more"* necessitates a Compare template. The model favored the temporal cue in the absence of a defined decision rule, overshadowing the interrogative cue.

**Case 2: Subtype code slip within Compare (GPT-4.1).** *Prompt:* "Mom has 12 flowers. She has 5 more flowers than Grandma. How many flowers does Grandma have?"

*Ground truth:* Compare—More, Smaller Quantity Unknown (CMSQU); roles: `bigger=12, difference=5, smaller=?`.

Model reasoning (summarized): GPT-4.1 accurately identified a Compare—More problem and indicated that the unknown quantity (Grandma's) is the smaller value, calculating $12 - 5 = 7$. It encoded the subtype label as CMLQU ("Larger Quantity Unknown").

This constitutes an error because: The verbal rationale aligns with CMSQU; however, the emitted subtype code is inconsistent. This constitutes a labeling or serialization error, not a reasoning error; the answer and role values were accurate, but the subtype received an incorrect score.

**Case 3: Non-conforming / "Unknown" output with high tokens (Claude Sonnet 4).** *Prompt:* "Kersti's blue pencil is 14 cm long. The green one is 4 cm shorter. How long is the green pencil?"

*Observed behavior:* The model returned `category=Unknown, subcategory=Unknown` and no usable mapping or answer, while consuming an unusually large number of tokens ($\sim$2308 total; $\sim$938 prompt, $\sim$1370 completion).

*Likely cause:* Loss of adherence to the required JSON-only schema (format drift), leading to output rejection and fallback to an "Unknown" placeholder.

**Takeaway.** The dominant errors reflect slot-value correctness. Nutikas selects the correct COMPS roles, but the numbers written into those roles sometimes violate the schema (e.g., treating change as a signed value instead of a required nonnegative magnitude, failing to aggregate multiple actions into a single change, or emitting a mismatched subtype label). Strengthening the output contract and adding lightweight validation/normalization (normalize sign for change, enforce non-null change, and sum multi-step changes) should address these issues without altering the underlying reasoning.

## 4.2 Tier 2: End-User Evaluation (RQ3)

**Students (think-aloud and SUS-Kids).** The two SEN pupils (ages 7–11, bilingual) each completed ∼30-minute think-aloud sessions (three to four problems) followed immediately by the adapted SUS-Kids questionnaire [33]. SUS scores were **62.5** and **75.0** (mean **68.8**), placing the group average at the ~68-point "average usability" benchmark [36] and indicating acceptable-to-good usability despite the small $N$.

Both pupils required a single neutral prompt during the first item ("Keep talking") but navigated independently thereafter. Across all tasks, 14 critical incidents were logged and coded to *Engagement* ($n = 5$), *Navigation friction* ($n = 5$), and *Reasoning & representation* ($n = 4$), with tags to COMPS stages: *read* ($n = 3$), *map* ($n = 7$), *compute* ($n = 4$).

Navigation frictions included: (i) interpreting comparison vocabulary (*fewer* vs. *less*) at the *map* stage; (ii) placing the question mark in the intended slot during grammar-story questions; (iii) initially attempting to *type* rather than *drag* numbers; and (iv) pressing *Next* to seek help rather than advance. Reasoning & representation issues included occasional "guessing" of numbers without resolving the schema, especially in combine-type problems.

Engagement indicators were consistently positive: pupils showed no signs of boredom, commented on ease of arithmetic, and one requested celebratory feedback ("Where are the balloons and music?"). On SUS-Kids engagement-only items (not part of SUS score), ratings were high (Pupil A: 4,5; Pupil B: 5,4), matching the positive think-aloud affect.

**Teacher (dashboard rubric and interview).** The teacher dashboard was scored **4/5** for Navigation, **3/5** for Analytics, and **N/A** for Language Accessibility (teacher-facing interface). Navigation comments noted that once set up, the interface was straightforward, but onboarding could be improved, particularly for class creation via CSV import.

Analytics clarity was the lowest-rated dimension: the "accuracy" score lacked an inline definition, and there was no breakdown between *conceptual* (schema-level) and *computational* (calculation) errors. In interview, the teacher stressed that this distinction is essential for targeted instructional responses. Suggested improvements included:

- Adding tooltips or help icons explaining metrics in plain language;
- Providing printable, low-scaffold task sets for differentiated use;
- Including a quick-start guide within the dashboard.

**Expert (ECA-SEN walkthrough).** The expert review of 12 sampled tasks (10% of Tier 1 corpus) yielded median ratings of **5/5** for *Linguistic Accessibility*, **5/5** for *Multimodal Alignment*, and **4/5** for *Cognitive Load & Scaffolds*, and **3/5** for *Logical Flow & Coherence*.

Strengths included short, concrete language, deliberate disambiguation of *fewer* vs. *less*, and exact correspondence between numbers, visuals, and text. Lower *Logical Flow* scores reflected small breaks in the reasoning chain (e.g., jumps between COMPS roles without explicit connectors), and a reliance on repeated *Next* presses rather than embedded causal links ("because... therefore...").

While scaffolding was generally effective, uniform repetition of certain symbols (e.g., question mark) occasionally encouraged procedural clicking over reflective reasoning.

**Converging evidence.** Table 5 synthesises the evidence across pupils, teacher, and expert review. Across all strands, **language accessibility** and **interface navigation** emerged as clear strengths. Priority improvement areas include:

- Enhancing *Logical Flow* in learner-facing explanations;
- Introducing variation in scaffold design to reduce procedural clicking;
- Adding inline explanations for teacher-facing metrics, with a clear separation of conceptual vs. computational errors.

| Theme | Pupils (Think-aloud + SUS-Kids) | Teacher (Dashboard) | Expert (ECA–SEN) |
|---|---|---|---|
| Engagement | High affect; celebratory feedback request; SUS-Kids engagement 4–5 | Positive observation of pupil motivation | N/A |
| Navigation | Independent after first item; SUS mean 68.8 (avg usability) | Navigation 4/5; CSV onboarding gap | Clear quiz-flow structure |
| Language Accessibility | No reading issues; comparison terms occasionally tricky | Supports SEN clarity | 5/5; consistent terminology |
| Conceptual vs. Computational Clarity | Guessing in combine-type problems; schema mapping errors | Metric does not distinguish types of error | Logical flow breaks (3–5/5) |
| Analytics Transparency | N/A (pupil side) | Accuracy metric unclear; no tooltip | N/A |
| Scaffolding | Some procedural clicking; symbol overuse | Suggests printable, low-scaffold tasks | 5/5 Cognitive Load; over-reliance on repeated elements |
| Multi-modal Alignment | Text/number/visuals aligned | N/A | 5/5; strong match between modalities |

**Table 5.** Summary of Tier 2 usability and instructional value evidence

# 5 Discussion and Conclusion

**RQ1 (technical accuracy).** The near-perfect classification accuracy across all models signals that the COMPS taxonomy is learnable and consistently applied by LLMs in early-grade additive contexts. The remaining classification slips; largely triggered by competing temporal and comparative cues, suggest that accuracy is not limited by capacity but by decision rule prioritization. This is encouraging: with refined cue-handling prompts or post-processing rules, those last percentage points are recoverable.

**RQ2 (equation fidelity and answer accuracy).** The ceiling-level arithmetic accuracy confirms that basic computation is not where improvement is needed. Instead, fidelity in representation i.e., slotting the right values into the right schema roles, is the true bottleneck. The Change–Separate polarity issue and the treatment of multi-action changes reveal that models still encode quantities in ways that make sense internally but violate the external pedagogical contract. This is a subtle but crucial distinction: in a teaching tool, how a solution is represented can matter as much as whether the final number is right. Simple guardrails (sign normalization, aggregation rules, no-null constraints) would likely fix most of these cases.

**RQ3 (usability).** The pilot data show that Nutikas is already accessible and motivating for SEN learners, with no major navigation barriers after minimal exposure. Engagement was genuine, not just procedural compliance, which bodes well for sustained classroom use. However, subtle usability gaps, like occasional guessing on combine problems and the lack of explicit conceptual bridges in explanations, could weaken the instructional value over time. On the teacher side, analytics clarity is the main concern. Without a clear split between conceptual and computational errors, teachers cannot target interventions effectively, limiting the practical impact of otherwise accurate AI output. Addressing this is as much a pedagogical requirement as a UX one.

**Efficiency note.** More tokens did not correspond to better mapping (e.g., the chattiest model was not the most faithful), reinforcing that the bottleneck is representational rather than about raw capacity.

## 5.1 Limitations and Threats to Validity

**Dataset and scope.** This evaluation covered only early-grade additive word problems (Part–Part–Whole and Additive Compare). Multiplicative structures, multi-step problems, and more linguistically complex items were not included, so generalisability beyond this domain is unknown. The dataset combined locally adapted special-education materials with ChatGPT-authored items to ensure subtype coverage; results were not stratified by source, so potential differences in difficulty or linguistic style remain unexamined.

**Annotation and scoring.** The Tier 2 usability and instructional quality findings relied on ratings (e.g., ECA–SEN rubric), which were gathered from a single expert without inter-rater reliability checks; this limits the generalisability of qualitative judgements about linguistic accessibility, cognitive load, and explanation flow.

**Usability pilot.** Usability and instructional value findings are based on a very small sample (two students with SEN, one expert reviewer, one teacher) and should be interpreted as formative signals rather than definitive evidence. The live pilot used a single model backend, so pedagogical comparisons across LLMs were not attempted. Observed behaviours (e.g., navigation ease, engagement, guessing) may not generalise to broader or more diverse learner populations.

## 5.2 Future Work

1. **Representation guardrails.** Implement automated checks and corrections for COMPS schema fidelity, sign normalisation ($\text{change} := |\Delta|$), aggregation of multi-action changes, and a no-null constraint for required fields, then re-score to measure the proportion of mapping errors eliminated.

2. **Analytics redesign.** Enhance teacher-facing reporting by separating conceptual (schema-level) and computational (calculation) errors, adding plain-language tooltips for metrics, and providing printable low-scaffold task sets for differentiation.

3. **Explanation flow improvements.** Make the link between narrative cues and COMPS slots more explicit in learner-facing explanations, and vary scaffold presentation to reduce procedural clicking while maintaining cognitive support.

4. **Extended coverage and robustness.** Broaden scope to multiplicative COMPS families (equal groups, rate, array) and two-step problems.

5. **Classroom-scale evaluation.** Run a semester-long, multi-class trial to validate real-world effectiveness and scalability.

## 5.3 Concluding Statement

This study shows that LLMs, when constrained by COMPS principles, can achieve near-ceiling classification and computational accuracy on early-grade additive word problems for SEN learners. The remaining technical gap lies in ensuring COMPS-conformant representation of problem structure. The usability pilot suggests that Nutikas is accessible and motivating for students, but that teacher-facing analytics require clearer conceptual/computational distinctions. Addressing these narrow, well-defined issues,

through representation guardrails, refined cue-handling, analytics redesign, and more explicit explanation flows, offers a realistic path from research prototype to scalable, classroom-ready AI tutor capable of delivering not just correct answers, but transparent, teachable reasoning at scale.

# References


[1] NCES. *2019 Mathematics State Snapshot Report: Nation Grade 4*. Nation Grade 4; NAEP Mathematics. 2019. URL: https://nces.ed.gov/nationsreportcard/subject/publications/stt2019/pdf/2020013np4.pdf.

[2] Jonté A. Myers et al. "A Meta-Analysis of Mathematics Word-Problem Solving Interventions for Elementary Students Who Evidence Mathematics Difficulties". In: *Review of Educational Research* 92.5 (2022). Statement on reading-dependent, word-problem-style NAEP items, pp. 695–742. DOI: 10.3102/00346543211070049.

[3] European Agency for Special Needs and Inclusive Education. *eBulletin November 2017: Research on Inclusive Education in Estonia*. "In 2014, there were almost 26,000 learners with SEN ... the share increased from 13.9% to 17.1–18.5%.". 2017. URL: https://www.european-agency.org/news/ebulletin-november-2017.

[4] Lieven Verschaffel et al. "Word problems in mathematics education: a survey". In: *ZDM* 52 (Apr. 2020), pp. 1–16. DOI: 10.1007/s11858-020-01130-4.

[5] David Jonassen. "Designing Research-Based Instruction for Story Problems". In: *Educational Psychology Review* 15 (Sept. 2003), pp. 267–296. DOI: 10.1023/A:1024648217919.

[6] Yan Ping Xin. *Conceptual Model-Based Problem Solving: Teach Students with Learning Difficulties to Solve Math Problems*. Rotterdam: Brill | Sense, 2012. ISBN: 978-94-6209-104-7. DOI: 10.1007/978-94-6209-104-7.

[7] Yan Ping Xin, Ben Wiles, and Yu-Ying Lin. "Teaching conceptual model–based word-problem story grammar to enhance mathematics problem solving". In: *The Journal of Special Education* 42.3 (2008), pp. 163–178. DOI: 10.1177/0022466907312895.

[8] Yan Ping Xin et al. "The Effect of Model-Based Problem Solving on the Performance of Students Who are Struggling in Mathematics". In: *The Journal of Special Education* 57.3 (2023), pp. 181–192. DOI: 10.1177/00224669231157032.

[9] Amy Lein, Asha Jitendra, and Michael Harwell. "Effectiveness of Mathematical Word Problem Solving Interventions for Students With Learning Disabilities and/or Mathematics Difficulties: A Meta-Analysis". In: *Journal of Educational Psychology* 112 (Jan. 2020). DOI: 10.1037/edu0000453.

[10] OpenAI, Josh Achiam, et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[11] Jason Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: 2201.11903 [cs.CL]. URL: https://arxiv.org/abs/2201.11903.

[12] Luyu Gao et al. *PAL: Program-aided Language Models*. 2023. arXiv: 2211.10435 [cs.CL]. URL: https://arxiv.org/abs/2211.10435.

[13] Joy He-Yueya et al. *Solving Math Word Problems by Combining Language Models With Symbolic Solvers*. 2023. arXiv: 2304.09102 [cs.CL]. URL: https://arxiv.org/abs/2304.09102.

[14] Photomath. *Photomath — The Ultimate Math Help App*. https://photomath.com/en. Accessed 9 Aug 2025. 2025.

[15] Apple App Store. *QANDA: AI Math & Study Helper*. https://apps.apple.com/us/app/qanda-ai-math-study-helper/id1270676408. Accessed 9 Aug 2025. 2025.

[16] TI-Hüpe. *Home*. 2025. URL: https://www.aileap.ee/en (visited on 08/12/2025).

[17] Bradley Witzel, Myers Jonte, and Yan Ping Xin. "Intensive Word Problem Solving for Students With Learning Disabilities in Mathematics". In: *Intervention in School and Clinic* 58 (Sept. 2021), p. 105345122110475. DOI: 10.1177/10534512211047580.

[18] Corey Peltier and Kimberly Vannest. "A Meta-Analysis of Schema Instruction on the Problem-Solving Performance of Elementary School Students". In: *Review of Educational Research* 87 (Oct. 2017), pp. 899–920. DOI: 10.3102/0034654317720163.

[19] Soo Kim and Yan Ping Xin. "A Meta-Analysis of Technology-Based Word-Problem Interventions for Students with Disabilities". In: *Education Sciences* 14 (Dec. 2024), p. 1372. DOI: 10.3390/educsci14121372.

[20] Shuang Wei, Qingli Lei, and Yan Ping Xin. "The Effects of Visual Cueing on Students with and without Math Learning Difficulties in Online Problem Solving: Evidence from Eye Movement". In: *Behavioral Sciences* 13 (Nov. 2023), p. 927. DOI: 10.3390/bs13110927.

[21] Yan Ping Xin et al. "The impact of a conceptual model-based mathematics computer tutor on multiplicative reasoning and problem-solving of students with learning disabilities". In: *The Journal of Mathematical Behavior* 58 (June 2020), p. 100762. DOI: 10.1016/j.jmathb.2020.100762.

[22] Yan Ping Xin et al. "Effect of Model-Based Problem Solving on Error Patterns of At-Risk Students in Solving Additive Word Problems". In: *Education Sciences* 13 (July 2023), p. 714. DOI: 10.3390/educsci13070714.

[23] Taekwon Son. “Intelligent Tutoring Systems in Mathematics Education: A Systematic Literature Review Using the Substitution, Augmentation, Modification, Redefinition Model”. In: *Computers* 13 (Oct. 2024), p. 270. DOI: 10.3390/computers13100270.

[24] Jingxi Liu et al. “Designing a Generative AI Enabled Learning Environment for Mathematics Word Problem Solving in Primary Schools: Learning Performance, Attitudes and Interaction”. In: *Computers and Education: Artificial Intelligence* 9 (June 2025), p. 100438. DOI: 10.1016/j.caeai.2025.100438.

[25] Ernest Davis. *Mathematics, word problems, common sense, and artificial intelligence*. 2023. arXiv: 2301.09723 [cs.AI]. URL: https://arxiv.org/abs/2301.09723.

[26] Pranab Sahoo et al. *A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications*. 2025. arXiv: 2402.07927 [cs.AI]. URL: https://arxiv.org/abs/2402.07927.

[27] *Math-Drills. Free Math Worksheets and Resources*. Accessed 2025-08-12. Math-Drills.com. URL: https://math-drills.com/ (visited on 08/12/2025).

[28] *Numberless Word Problems. Problem Banks*. Accessed 2025-08-12. Numberless Word Problems. URL: https://numberlesswp.com/problem-banks/ (visited on 08/12/2025).

[29] *Math Word Problems*. Accessed 2025-08-12. Prodigy Education. URL: https://www.prodigygame.com/main-en/blog/math-word-problems/ (visited on 08/12/2025).

[30] Chioma Nkem-Eze. *Nutikas Additive Word-Problem Evaluation Set (PPW & AC), v1.0*. Version 35133f459d58cf31ce1d2d3fb447ca1e15067ba0. Frozen GitHub folder; 120 items with gold COMPS labels and scoring schema. 2025. URL: https://github.com/fegaeze/ai_special_education_thesis/tree/35133f459d58cf31ce1d2d3fb447ca1e15067ba0/ai_evaluator/data (visited on 08/12/2025).

[31] Chioma Nkem-Eze. *ai_special_education_thesis: Nutikas code, prompts, and materials*. GitHub repository. 2025. URL: https://github.com/fegaeze/ai_special_education_thesis (visited on 08/12/2025).

[32] OpenRouter. *Model Leaderboard and Provider Comparison*. Accessed for model quality/ranking guidance. 2025. URL: https://openrouter.ai/.

[33] Cynthia Putnam et al. “Adaptation of the System Usability Scale for User Testing with Children”. In: *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI EA ’20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1–7. ISBN: 978-1-4503-6819-3. DOI: 10.1145/3334480.3382840.

[34] CAST. *Universal Design for Learning Guidelines version 3.0*. Tech. rep. CAST, Inc., 2024. URL: https://udlguidelines.cast.org/.

[35] Smarter Balanced. *Performance Task Writing Rubric* — Explanatory *(Grades 6–11)*. Tech. rep. Smarter Balanced Assessment Consortium, 2022. URL: https://portal.smarterbalanced.org/library/en/performance-task-writing-rubric-explanatory.pdf.

[36] Aaron Bangor, Philip Kortum, and James Miller. "An Empirical Evaluation of the System Usability Scale". In: *International Journal of Human–Computer Interaction* 24.6 (2008), pp. 574–594. DOI: 10.1080/10447310802205776.

# Appendix

## .1 Super-Category Classification Prompt

```
You are a math educator trained in the **COMPS (Conceptual Model-Based Problem
    Solving)** framework.
Your task is to classify each arithmetic word problem into one of the following
    conceptual categories:

## Change
Describes a situation where a **quantity increases or decreases over time** due to
    some event (e.g., gaining, losing, spending, receiving).
It involves a **transformation** - a quantity starts at one amount and becomes a
    different amount.
These problems typically follow a **before-and-after structure** or unfold across
    **multiple time points**.
### Common Indicators
- Actions like: _gave_, _lost_, _got_, _added_, _received_, _left_, _joined_
- Tense or language indicates **sequence** or **time passing**

## Combine
Describes a situation where **two or more distinct quantities** are **grouped
    together into a total**.
There is **no transformation** - nothing changes over time. The parts are simply
    added to find a total.
Even if events happened at different times (e.g., two races), if there is no **
    change in state**, it's **Combine**.
### Common Indicators
- Parts are distinct but **independent**
- No item is being gained/lost/transferred
- Words like: _in total_, _altogether_, _sum_

## Compare
Describes a situation where **two quantities are contrasted** to find the **
    difference** between them
(or determine which is more, less, or how much more or less).
### Common Indicators
- Questions asking: _how many more..._, _how many fewer..._
- Focus is on **difference**, **not change** or **combining**

---

### Classification Guide

Think step-by-step:
1. **Does the problem describe a quantity that changes over time**
   due to an action like gaining or losing?
   -> **Change**
```

```
2. **Does the problem group together multiple distinct quantities**
   without affecting each other?
   -> **Combine**
3. **Does the problem contrast two amounts** to find a difference or comparison?
   -> **Compare**
> Focus on what the story is **doing**: changing something, grouping parts, or
    contrasting values.

---

### Output Format
Return your answer in the following JSON array format:

[
  {{
    "problem": "<problem text>",
    "category": "Change|Combine|Compare",
    "reasoning": "<explanation of your reasoning in full sentences and connected
        thoughts>"
  }}
]


Here are the problems:
<one problem per line...>
```

## .2 Sub-Category Classification Prompt

### Change..

```
You are classifying this word problem using the COMPS (Conceptual Model-Based
    Problem Solving) framework.

Classify it into one of the following types:
- CJPU: Change-Join, Part Unknown
- CJWU: Change-Join, Whole Unknown
- CSPU: Change-Separate, Part Unknown
- CSWU: Change-Separate, Whole Unknown

---

### Step-by-Step Instructions

#### Step 1: Determine the Situation Type
Is the quantity increasing or decreasing over time?
- If it **increases** -> it's a **Join** problem
- If it **decreases** -> it's a **Separate** problem

---

### Step 2: Identify the Quantities in the Model
Use this structure:
Start +/- Change = Result

Now fill in each of the three quantities from the problem:
- **Start**: the initial amount before anything is added or removed
- **Change**: the amounts that are added (Join) or removed (Separate)
- **Result**: the final amount after the change has happened
Label whichever value is unknown as "unknown".

---

### Step 3: Identify the Unknown and Its Role

#### For **Join**:
- Start and Change are **Parts**, Result is the **Whole**
- If **Start** or **Change** is unknown -> it's a **Part Unknown** -> **CJPU**
- If **Result** is unknown -> it's a **Whole Unknown** -> **CJWU**

#### For **Separate**:
- Start is the **Whole**, Change and Result are **Parts**
- If **Start** is unknown -> it's a **Whole Unknown** -> **CSWU**
- If **Change** or **Result** is unknown -> it's a **Part Unknown** -> **CSPU**

> Don't just label based on what is missing.
> Classify based on what that missing value **represents** in the model.
```

```
Do not rephrase or restate the problem in a different form.
Always classify based on the original sentence structure and comparison direction.

Problem:
<one problem per line...>

Respond strictly in the following JSON format with no extra commentary:
{
  "reasoning": "<explanation of your reasoning in full sentences and connected
      thoughts>",
  "subcategory": "CJPU | CJWU | CSPU | CSW"
}
```

**Combine..**

```
You are classifying this word problem using the COMPS (Conceptual Model-Based
    Problem Solving) framework.

Classify it into one of the following types:
- **CPU**: Combine, Part Unknown
- **CWU**: Combine, Whole Unknown

---

### Step-by-Step Instructions

#### Step 2: Identify the Quantities in the Combine Model

Use this structure:
> Part 1 + Part 2 = Whole

Now fill in from the problem:
- **Part 1**: a known amount in the group
- **Part 2**: another known amount in the group
- **Whole**: the total combined amount
Mark whichever one is **unknown**.

#### Step 3: Determine the Type of Unknown
- If one of the **Parts** is unknown  it's a **Part Unknown**  **CPU**
- If the **Whole** is unknown  it's a **Whole Unknown**  **CWU**

Do not rephrase or restate the problem in a different form.
Always classify based on the original sentence structure and comparison direction.

Problem:
<one problem per line...>

Respond strictly in the following JSON format with no extra commentary:
```

```
{
  "reasoning": "<explanation of your reasoning in full sentences and connected
      thoughts>",
  "subcategory": "CPU | CWU"
}
```

### Compare..

```
You are classifying this word problem using the COMPS (Conceptual Model-Based
    Problem Solving) framework.

Choose one of:
- CLDU, CLLQU, CLSQU (Compare-Less)
- CMDU, CMLQU, CMSQU (Compare-More)

Classify it into one of the following types:

## Compare-Less
- **CLDU** - Compare-Less, Difference Unknown
- **CLLQU** - Compare-Less, Larger Quantity Unknown
- **CLSQU** - Compare-Less, Smaller Quantity Unknown

## Compare-More
- **CMDU** - Compare-More, Difference Unknown
- **CMLQU** - Compare-More, Larger Quantity Unknown
- **CMSQU** - Compare-More, Smaller Quantity Unknown

---

## Step-by-Step Instructions

1. **Determine the Comparison Direction**
   - Is the problem asking **how many more**, **how many fewer**, or comparing one
       group to another?
   - If the focus is on **"less than"**, it's a **Compare-Less** problem.
   - If the focus is on **"more than"**, it's a **Compare-More** problem.

2. **Use the Compare Model**
   According to the COMPS framework:
   > Bigger = Smaller + Difference

   Now fill in from the problem:
    - **Bigger**: larger quantity
    - **Smaller**: smaller quantity
    - **Difference**: how much more or less
   Mark whichever one is unknown.

3. **Identify the Unknown**
Which of the three quantities is missing?
```

```
- **Difference Unknown** -> *_DU
- **Larger Quantity Unknown** -> *_LQU
- **Smaller Quantity Unknown** -> *_SQU

Do not rephrase or restate the problem in a different form.
Always classify based on the original sentence structure and comparison direction.

Problem:
<one problem per line...>

Respond strictly in the following JSON format with no extra commentary:
{
  "reasoning": "<explanation of your reasoning in full sentences and connected
      thoughts>",
  "subcategory": "CLDU | CLLQU | CLSQU | CMDU | CMLQU | CMSQU"
}
```

## .3 Schema Mapping Prompt

### Change..

```
You are solving a **Change** word problem using the COMPS model:
> Start +/- Change = End

---

## Step-by-Step Guide
1. Determine if the problem describes an **increase** (Join) or **decrease** (
    Separate) in quantity.
2. Identify the three model parts:
   - **Start**: the amount before anything happened
   - **Change**: the amount added or taken away
   - **End**: the result after the change

---

## Your Task
1. Extract the values from the story.
2. Assign each number to the correct model label: start, change, end.
3. If a value is unknown, write null.
4. Calculate the missing value, and assign it to answer.
5. Explain your steps clearly.

Example Output:
{{
  "modelAnswers": {{
    "start": 12,
    "change": null,
    "end": 18
  }},
  "answer": 6,
  "reasoning": "12 + ? = 18 -> 18 - 12 = 6, so the change is 6."
}}

IMPORTANT: All values in modelAnswers must be numbers or null. Do not include
    mathematical expressions like "2 + 2" - calculate the result and write "4"
    instead.

Problem:
<one problem per line...>

Only return the JSON object.
```

### Combine..

```
You are solving a **Combine** word problem using the COMPS model:
> Part 1 + Part 2 = Whole
```

```
---

## Step-by-Step Guide

1. Identify the three values:
   - **Part 1**: the first part of the group
   - **Part 2**: the second part of the group
   - **Whole**: the total after combining

---

## Your Task
1. Assign values to part1, part2, and whole.
2. Use "null" for the missing part.
3. Calculate the answer.
4. Explain your math clearly.

Example Output:

{{
  "modelAnswers": {{
    "part1": 4,
    "part2": 6,
    "whole": null
  }},
  "answer": 10,
  "reasoning": "4 + 6 = 10, so the total is 10."
}}

IMPORTANT: All values in modelAnswers must be numbers or null. Do not include
    mathematical expressions like "2 + 2" - calculate the result and write "4"
    instead.

Problem:
<one problem per line...>

Respond strictly in the following JSON format with no extra commentary:
{
  "reasoning": "<explanation of your reasoning in full sentences and connected
      thoughts>",
  "subcategory": "CPU | CWU"
}
```

### Compare..

```
You are solving a **Compare** word problem using the COMPS model:
> Bigger = Smaller + Difference
```

```
---

## Step-by-Step Guide

1. Identify the quantities:
   - **Bigger**: the larger amount
   - **Smaller**: the lesser amount
   - **Difference**: how much more or fewer

---

## Your Task

1. Assign values to bigger, smaller, and difference.
2. Use "null" for the unknown one.
3. Solve for the answer and explain how.

Example Output:

{{
  "modelAnswers": {{
    "bigger": null,
    "smaller": 4,
    "difference": 2
  }},
  "answer": 6,
  "reasoning": "Bigger = 4 + 2 = 6, so the bigger amount is 6."
}}

IMPORTANT: All values in modelAnswers must be numbers or null. Do not include
    mathematical expressions like "2 + 2" - calculate the result and write "4"
    instead.

Problem:
<one problem per line...>

Respond strictly in the following JSON format with no extra commentary:
{
  "reasoning": "<explanation of your reasoning in full sentences and connected
      thoughts>",
  "subcategory": "CLDU | CLLQU | CLSQU | CMDU | CMLQU | CMSQU"
}
```

## .4 Story Grammar Generation Prompt

### Schema Information.

```
   {
   CJPU: { model: "Start + Change = End", boxes: ["start", "change", "end"] },
   CJWU: { model: "Start + Change = End", boxes: ["start", "change", "end"] },
   CSPU: { model: "Start - Change = End", boxes: ["start", "change", "end"] },
   CSWU: { model: "Start - Change = End", boxes: ["start", "change", "end"] },
   CPU: { model: "Part1 + Part2 = Whole", boxes: ["part1", "part2", "whole"] },
   CWU: { model: "Part1 + Part2 = Whole", boxes: ["part1", "part2", "whole"] },
   CMDU: {
     model: "Bigger = Smaller + Difference",
     boxes: ["bigger", "smaller", "difference"],
   },
   CMLQU: {
     model: "Bigger = Smaller + Difference",
     boxes: ["bigger", "smaller", "difference"],
   },
   CMSQU: {
     model: "Bigger = Smaller + Difference",
     boxes: ["bigger", "smaller", "difference"],
   },
   CLDU: {
     model: "Bigger = Smaller + Difference",
     boxes: ["bigger", "smaller", "difference"],
   },
   CLLQU: {
     model: "Bigger = Smaller + Difference",
     boxes: ["bigger", "smaller", "difference"],
   },
   CLSQU: {
     model: "Bigger = Smaller + Difference",
     boxes: ["bigger", "smaller", "difference"],
   },
 };
```

### Prompt with data from Schema Information.

```
   You are an expert elementary math teacher helping a student solve a word problem
        using the COMPS (Conceptual Model-Based Problem Solving) framework.

   The student is in Grades 1-4 and needs clear, supportive help understanding what
        to do at each step.

---

Problem:
<one problem per line...>
```

```
Subtype: ${subtype}
Model Type: ${modelInfo.model}
Equation Parts: ${modelInfo.boxes.join(", ")}
Model Answers: ${modelAnswers}
Final Answer: ${answer}

---

Your Task

Generate a list of ${modelInfo.boxes.length} questions - one for each model part -
    that will help the child fill in the model equation.

For each part, generate:

- **text**: a simple story grammar question in the student's language
- **boxTarget**: the model box it refers to (e.g., "start", "whole")
- **context**: a friendly explanation of what the story grammar question asks,
    using warm, simple language for young readers in grades 1-4.

Keep the tone warm and supportive, as if you're sitting next to the child.
Do **not** rephrase the problem. Do **not** give away the answer.

---

Output Format:
[
  {{
    "text": "Story grammar question for the student",
    "boxTarget": "start",
    "context": "a friendly explanation of what the story grammar question asks,
        using warm, simple language for young readers in grades 1-4."
  }},
  ...
]

Use clear and natural language. Do not use math words like "variable" or "term".
    Use the actual items from the story where it helps.

Now write the JSON array.
```

## .5 COMPS Additive Subtype Groups

| Super-category | Subtype codes |
|---|---|
| Change | Change–Join, *Part Unknown* (CJPU) |
| | Change–Join, *Whole Unknown* (CJWU) |
| | Change–Separate, *Part Unknown* (CSPU) |
| | Change–Separate, *Whole Unknown* (CSWU) |
| Combine | Combine, *Part Unknown* (CPU) |
| | Combine, *Whole Unknown* (CWU) |
| Compare | Compare–Less, *Difference Unknown* (CLDU) |
| | Compare–Less, *Larger Quantity Unknown* (CLLQU) |
| | Compare–Less, *Smaller Quantity Unknown* (CLSQU) |
| | Compare–More, *Difference Unknown* (CMDU) |
| | Compare–More, *Larger Quantity Unknown* (CMLQU) |
| | Compare–More, *Smaller Quantity Unknown* (CMSQU) |

**Table 6.** COMPS additive subtypes grouped by super-category.

## .6 COMPS Subtype Decision Cards

### .6.1 Change Decision Card

**Schema:** Start $\pm$ Change $=$ End.

| Change case | Subtype rule (by unknown role) |
|---|---|
| **Join (increase)** | Start and Change are *parts*; End is the *whole*.<br>Unknown Start or Change $\Rightarrow$ `CJPU`; Unknown End $\Rightarrow$ `CJWU`. |
| **Separate (decrease)** | Start is the *whole*; Change and End are *parts*.<br>Unknown Start $\Rightarrow$ `CSWU`; Unknown Change or End $\Rightarrow$ `CSPU`. |

**Table 7.** COMPS Change case and decision rule.

### .6.2 Combine Decision Card

**Schema:** $\text{Part}_1 + \text{Part}_2 =$ Whole.

| Combine case | Subtype rule (by unknown role) |
|---|---|
| **Combine (no temporal change)** | $\text{Part}_1$ and $\text{Part}_2$ are *parts*; Whole is the *total*.<br>Unknown Whole $\Rightarrow$ `CWU`; Unknown $\text{Part}_1$ or $\text{Part}_2$ $\Rightarrow$ `CPU`. |

**Table 8.** COMPS Combine case and decision rule.

### .6.3 Compare Decision Card

**Schema:** Bigger $=$ Smaller $+$ Difference.

| Compare case | Subtype rule (by unknown role) |
|---|---|
| **Compare–More** | Use original wording to set direction (“more than”); map with Bigger = Smaller + Difference.<br>Unknown Difference ⇒ `CMDU`; Unknown Bigger ⇒ `CMLQU`; Unknown Smaller ⇒ `CMSQU`. |
| **Compare–Less** | Use original wording to set direction (“less/fewer than”); map with Bigger = Smaller + Difference.<br>Unknown Difference ⇒ `CLDU`; Unknown Bigger ⇒ `CLLQU`; Unknown Smaller ⇒ `CLSQU`. |

**Table 9.** COMPS Compare cases and decision rules.

## .7 Dataset Brief

**Locations (frozen at commit)..** All files used for Tier 1 evaluation are pinned at commit `35133f459d58cf31ce1d2d3fb447ca1e15067ba0` of the project repository.

- **Evaluation Input Dataset** `ai_evaluator/data/`
  - Balanced set of 120 additive word problems (10 per COMPS subtype across PPW/AC)
- **Evaluation Output Dataset** `server/seed/`
  - Per-item LLM predictions (category, subtype, modelAnswers, answer, rationale)
  - Ground-truth echo for each item and pass/fail flags (mapping/answer)
- *Not used in scoring (for provenance only):* `old/testing_data`, `old/training_data` (prompt-development pools; excluded from Tier 1)

**Model-output record schema (evaluation output).** Each scored record in `server/seed/` follows:

```
{
  "modelName": "Anthropic: Claude Sonnet 4",
  "problem": "5 tables were brought ... How many tables were there before?",
  "output": "free-text rationale for super-category",
  "category": "Change",                 // predicted super-category
  "subcategory": "CJPU",                // predicted subtype (12-way)
  "subReasoning": "free-text rationale for subtype",
  "tokenUsage": { "prompt": 452, "completion": 227, "total": 679 },

  // category-specific mapping (numbers or null only)
  "modelAnswers": { "start": null, "change": 5, "end": 15 },
  "answer": 10,                         // predicted numeric answer
  "reasoning": "one-sentence computation explanation",

  // ground truth + checks used for scoring
  "groundTruthCategory": "Change",
  "groundTruthSubcategory": "CJPU",
  "groundTruthAnswer": 10,
  "groundTruthModelAnswers": { "start": null, "change": 5, "end": 15 },
  "isModelMappingCorrect": true,
  "isAnswerCorrect": true
}
```

## .8 SEN-friendly think-aloud protocol

### .8.1 Preparation (10 minutes)

- Conduct the session in a quiet, familiar space (preferably the pupil's regular learning area).
- Provide a simple instruction sheet or read aloud:

  > "I want you to talk out loud about what you're thinking as you use the app to solve each math problem. Say whatever comes into your head—there are no right or wrong words."

- Demonstrate with a non-math example (e.g., finding a toy on a screen) and model self-talk: "I'm clicking here because…" so the pupil understands the idea of verbalising thought processes.

### .8.2 Visual supports for SEN accessibility

- Green cue-card ( **Talk**) – pupil holds when speaking.
- Red cue-card ( **Stop**) – pupil holds if they need a break or are stuck.
- Timer icon – shown after $\sim$2 minutes of thinking aloud to remind them they can pause or request a "wiggle break".

### .8.3 Session script (per pupil, $\sim$20 minutes)

1. **Warm-up** (2 minutes) Ask the pupil to describe out loud something trivial (e.g., "Tell me what you're thinking as you brush your teeth.") to practise continuous talking.
2. **Task rounds** (3 problems $\times$ 4 minutes each = 12 minutes)
   (a) Present the word problem in the Nutikas app.
   (b) Prompt: "Let me hear everything you're thinking as you read and solve it."
   (c) Sit slightly to the side; do not point or suggest answers. If silent for $>$10 s: "Keep talking!"
3. **Short debrief after each problem** (1 minute each = 3 minutes)

   > "What did you find easy? What was hard?"

4. **General wrap-up** (3 minutes)

"If you could change one thing about the app to make it better, what would it be?"

Note any body-language cues (frustration, delight, hesitation).

### .8.4 SEN adaptations

- Breaks on demand: pupil may hold the  card at any time to pause for up to 30 seconds.
- Pre-written prompt scripts in simple language, e.g.: "Can you tell me why you tapped that blue button?" "What are you thinking now?"
- Chunk timing: sessions chunked to avoid overload, with planned pauses between problems.

### .8.5 Critical incident observation sheet template

| **Time-stamp** | **Task ID** | **Incident description** | **Code** (Engage-ment / Navigation / Reasoning) | **COMPS stage** (Read / Map / Compute) | **Severity** (1–3) |
|---|---|---|---|---|---|
| 00:03 | Q1 | Hesitated on "fewer" | Reasoning | Map | 2 |
| 00:05 | Q1 | Clicked "Next" for help | Navigation | Compute | 1 |

**Table 10.** Template for logging think-aloud critical incidents.

## .9 Adapted SUS-Kids instrument and scoring

Two SEN pupils (Pupil A and Pupil B) completed the 12-item adapted SUS-Kids questionnaire using a 5-point smiley-face Likert scale (1 = strongly disagree; 5 = strongly agree). Items included both positively and negatively worded statements about the usability and enjoyment of the Nutikas app. Table 11 shows the raw scores for each item.

| # | Item | Pupil A | Pupil B |
|---|---|---|---|
| 1 | I would like to play Nutikas a lot more. | 4 | 5 |
| 2 | Nutikas was hard to play. | 3 | 4 |
| 3 | I thought Nutikas was easy to use. | 4 | 5 |
| 4 | I would need help to play Nutikas more. | 2 | 1 |
| 5 | I knew what to do next when I played Nutikas. | 5 | 5 |
| 6 | Some things in Nutikas made no sense. | 3 | 3 |
| 7 | Nutikas would be easy for my friends to learn. | 1 | 5 |
| 8 | To play Nutikas I had to do some weird things. | 2 | 3 |
| 9 | I was proud of how I played Nutikas. | 5 | 3 |
| 10 | There was a lot to learn to play Nutikas. | 4 | 2 |
| 11 | Playing Nutikas was fun. | 4 | 5 |
| 12 | I plan on telling my friends about Nutikas. | 5 | 4 |

**Table 11.** Raw SUS-Kids scores for two SEN pupils (1 = strongly disagree, 5 = strongly agree).

**Scoring procedure..** For SUS-style scoring, only items 1–10 are used. Positively worded items (odd-numbered) are scored as $(\text{raw} - 1)$, negatively worded items (even-numbered) as $(5 - \text{raw})$. The adjusted scores are summed (range 0–40) and multiplied by 2.5 to produce a score out of 100.

**Results..**

- **Pupil A:** SUS = 62.5 — slightly below the conventional "average usability" threshold of 68.
- **Pupil B:** SUS = 75.0 — above average, indicating strong perceived usability.

Both pupils reported that they knew what to do next, could use the app largely without help, and enjoyed the experience (Items 11–12). Engagement scores were high (Pupil A: 4, 5; Pupil B: 5, 4), aligning with think-aloud observations that both pupils were more engaged than with traditional materials.

The lowest score for Pupil A was *"My friends could learn it easily"* (score 1), suggesting that while the app felt usable personally, it may feel more challenging for peers with different needs.

Overall, these results are encouraging given the small sample size and SEN context, with Pupil B consistently more positive across items.

## .10 Explanation Coherence and Accessibility Rubric (ECA–SEN)

The ECA–SEN rubric evaluates model-generated explanations across four dimensions:

1. **Logical Flow & Coherence** — clarity, order, and thematic unity of ideas.
2. **Linguistic Accessibility** — plain language, sentence length, and vocabulary suitability for SEN learners.
3. **Cognitive Load & Scaffold** — use of chunking, icons, and step-by-step support to minimise cognitive demands.
4. **Multi-modal Alignment** — consistency between text, visuals, and other modes.

Each dimension is scored on a 1–5 scale (1 = very poor, 5 = excellent), with detailed descriptors for each score level (Table 12).

| Dimension | 5 — Excellent | 4 — Good | 3 — Adequate | 2 — Poor / 1 — Very Poor |
|---|---|---|---|---|
| **Logical Flow & Coherence** | Clear beginning–middle–end, explicit transitions, no digressions. | Minor lapses (e.g., one weak transition) but overall easy to follow. | Occasional missing links; reader must infer once or twice. | Disordered or illogical steps; reader cannot construct mental model. |
| **Linguistic Accessibility** | All sentences $\leq$ 15 words, everyday vocabulary, zero jargon. | 1–2 sentences over 20 words; one mild abstract term. | Several long sentences or unexplained Tier-2 words. | Frequent complex sentences or unfamiliar vocabulary throughout. |
| **Cognitive Load & Scaffold** | Fully chunked steps, headings, bullets, supportive visuals/icons. | Mostly chunked; one information-dense segment. | Long paragraph or missing icon may tax working memory. | Dense block text, no visuals, high risk of overload. |
| **Multi-modal Alignment** | Perfect match between text and visuals (labels, colour coding). | Minor mismatch, meaning still clear. | Partial misalignment (e.g., unlabeled arrow). | Contradictory or missing visuals. |

**Table 12.** ECA–SEN rubric scoring criteria (1–5 scale).

### .10.1 Teacher Ratings and Observations

Based on teacher evaluation of the Nutikas app explanations for SEN pupils:

- **Logical Flow & Coherence: 3** — The connection between conceptual and arithmetic steps was sometimes unclear. For example, the grammar story section could display "Great! You've completed all the story grammar prompts" even when the model equation was incomplete or incorrect. A corrective prompt earlier in the process would prevent pupils from continuing with an invalid equation.

- **Linguistic Accessibility: 5** — Clear, age-appropriate, and accessible to SEN pupils; vocabulary is simple and instructions are easy to follow.
- **Cognitive Load & Scaffold: 4** — Scaffolded well overall, but repeated "Next" clicks may lead to distraction or fatigue. The question mark symbol for the unknown could be replaced with a direct prompt such as "Find the missing number" to further reduce ambiguity.
- **Multi-modal Alignment: 5** — Strong alignment between text, visuals, and interactions. Additional modes (e.g., audio) could be explored, but the current setup is coherent and effective for the intended audience.

## .11 Teacher dashboard analytic rubric and interview results

### .11.1 Materials

Teachers evaluated the Nutikas dashboard using the **Teachers Analytic Rubric** (Appendix .11.3), which rates five key dimensions on a 5-point scale, with assigned weights for scoring. Each dimension contains descriptors for Very Poor (1) through Excellent (5) performance.

In addition, teachers participated in a **Semi-Structured Interview** (Appendix .11.4) designed to elicit qualitative feedback linked to their rubric ratings. The interview explored navigation, assignment control, analytics clarity, actionable feedback, and pedagogical alignment.

1. Analytic rubric (quantitative scores by dimension, weighted means).
2. Semi-structured interview prompts with targeted follow-ups.
3. Observations on feature usability and alignment with COMPS pedagogy.

### .11.2 Results

| Dimension | Mean score | SD | % $\geq 4$ | n | Representative comments |
|---|---|---|---|---|---|
| Navigation & Efficiency | 4.0 | – | 100% | 1 | Navigating from the class view to an individual pupil's work is straightforward. Would be useful to show a full list of completed tests per student rather than only general scores. Sorting and date-filter features exist but are underdeveloped. |
| Assignment Control | 4.5 | – | 100% | 1 | Creating a new quiz is easy. Suggestion: add an option to print problems (with model equations, without grammar stories) for in-class or homework use, to assess performance with less scaffolding. |
| Analytics Clarity & Accuracy | 3.0 | – | 0% | 1 | The "accuracy" score is unclear; no immediate explanation of what it represents. Teachers want a clear description of its pedagogical meaning and calculation method. |
| Actionable Feedback | 4.0 | – | 100% | 1 | Helps identify problem types the class struggles with, but more insight into whether errors are conceptual or arithmetical is desired. |
| Pedagogical Alignment | 5.0 | – | 100% | 1 | Excellent alignment with COMPS pedagogy, offering clear advantages over traditional keyword-based instruction. |

**Table 13.** Teacher analytic rubric results with representative qualitative comments (n = 1 teacher).

Overall, teachers rated the dashboard highly for assignment control, pedagogical alignment, and ease of navigation. Key suggestions included clearer analytics definitions, expanded error-type breakdowns, and minor improvements to navigation features.

### .11.3 Teachers Analytic Rubric

| Dimension | Weight | 1 — Very Poor | 2 — Poor | 3 — Adequate | 4 — Good | 5 — Excellent |
|---|---|---|---|---|---|---|
| Navigation & Efficiency | ×1 | Frequent dead ends; >5 clicks to reach core data; no breadcrumb. | Path discoverable but takes 4–5 clicks; trial-and-error. | Main tasks in 3 clicks; layout predictable but cluttered. | Most tasks in 2 clicks; clear labels, backtrail; minor redundancy. | Core tasks in 1–2 clicks; logical grouping, breadcrumb, no wasted steps. |
| Assignment Control | ×1 | Cannot assign tasks or adjust due dates; only default set. | Can create set but no schema/difficulty control; due date editable. | Choose schema or difficulty; due date & hints toggle available. | Fine-grain control over schema, difficulty, due date, hints; bulk-assign. | Full control, save templates, preview pupil view, schedule release. |
| Analytics Clarity & Accuracy | ×2 | Percent-correct unclear; graphs misleading or wrong totals. | Percent-correct visible but undefined; confusing scales. | Definition tooltip; correct item counts; limited drill-down. | Drill-down to item level; error bars; reliable live sync. | Transparent formulae, colour-blind-safe graphs, exportable CSV; anomaly flags. |
| Actionable Feedback | ×1 | No reteaching guidance; raw scores only. | Generic "Needs help" labels; no schema hints. | Text hints; printable worksheet link. | Schema-specific misconception tags; reteach tips; regroup pupils. | Adaptive suggestions keyed to each error; exemplar explanations. |
| Pedagogical Alignment | ×1 | Conflicts with COMPS terms; incorrect/missing models. | Neutral terms but inconsistent with pupil view. | Correct COMPS labels; visuals match pupil app. | Full COMPS language, colour coding, consistent model icons. | Adds COMPS theory notes, sample grammar stories, pacing cues. |

**Table 14.** Teachers Analytic Rubric for Nutikas dashboard evaluation.

### .11.4 Semi-Structured Interview Guide

1. **Navigation & Efficiency** "You rated Navigation # /5. Can you describe the moment you felt slowed down?" Follow-up: "Where would you expect that button to live?"

2. **Assignment Control** "You rated Assignment # /5. Tell me about creating a new set. What controls were missing for you?" Follow-up: "If we added just one more option, which would help most?"

3. **Analytics Clarity & Accuracy** "You rated Analytics Clarity # /5. Looking at Pupil A's accuracy graph, do you trust the number? Why or why not?" Follow-up: "What extra context would give you full confidence?"

4. **Actionable Feedback** "You rated Feedback # /5. How useful were the error explanations for planning tomorrow's lesson?" Follow-up: "Can you recall a time feedback actually changed your teaching decision?"

5. **Pedagogical Alignment** "You rated Pedagogical Alignment # /5. Does the dashboard's language match how you teach COMPS in class?" Follow-up: "Any terminology you would rephrase to fit your practice?"

6. **Closing** "If we could improve one thing before September, what would move the needle the most?" Follow-up: "Anything else you'd like to add that we haven't covered?"

## .12 Critical incident log (Tier~2 think-aloud sessions)

| Time | Pupil ID | Task ID | Incident type | COMPS stage | Severity | Notes |
|---|---|---|---|---|---|---|
| 00:02 | P03 | Q5 | Navigation error | Read | 2 | Opened unrelated menu before starting to read; recovered without assistance. |
| 00:05 | P03 | Q5 | Reasoning hesitation | Map | 3 | Stopped mid-sentence, unsure how "more than" related to numbers given. |
| 00:08 | P03 | Q5 | Engagement drop | Compute | 2 | Looked away from screen; resumed after prompt "Keep talking". |
| 00:04 | P07 | Q2 | Misinterpretation | Read | 3 | Read question incorrectly, reversed roles of quantities; required re-reading. |
| 00:06 | P07 | Q2 | Accessibility barrier | Map | 1 | Confused by term "fewer"; guessed instead of clarifying meaning. |
| 00:03 | P11 | Q8 | Calculation slip | Compute | 2 | Performed subtraction correctly but entered wrong answer due to keypress error. |
| 00:07 | P11 | Q8 | Engagement cue | Map | 1 | Smiled and commented positively on problem theme; maintained task focus. |

**Table 15.** Example critical incidents logged during Tier 2 think-aloud sessions. Severity: 1 = low impact, 3 = high impact.

# Acronyms

| | |
|---|---|
| AC | Additive Compare. |
| AI | artificial intelligence. |
| API | application programming interface. |
| CAI | computer-assisted instruction. |
| COMPS | Conceptual Model-Based Problem Solving. |
| ECA–SEN | Expert Conceptual Accessibility rubric for SEN. |
| EG | Equal Groups. |
| IQR | interquartile range. |
| ITS | intelligent tutoring system. |
| JSON | JavaScript Object Notation. |
| LLM | large language model. |
| NAEP | National Assessment of Educational Progress. |
| ORM | object–relational mapper. |
| PPW | Part–Part–Whole. |
| REST | representational state transfer. |
| SBI | schema-based instruction. |
| SEN | special educational needs. |
| SUS | System Usability Scale. |
| UI | user interface. |

# Glossary

| | |
|---|---|
| Bigger | The larger of two compared quantities (Compare). |
| chain-of-thought | A prompting technique that elicits step-by-step intermediate reasoning before the final answer. |
| Change | The nonnegative magnitude that is added to or removed from the start amount (Change). |
| Change | Situations with a before/after state where a quantity increases (join) or decreases (separate). |
| Combine | Part–Part–Whole situations where two or more parts compose a whole. |
| Compare | Situations that contrast two quantities via a difference. |
| computational error | Arithmetic mistake given a correct schema and correct numeral-to-slot mapping. |
| conceptual misclassification | Assigning the wrong super-category (Change/-Combine/Compare) or wrong COMPS subtype. |
| Difference | How much larger one quantity is than the other (Compare). |
| End | The amount after the change occurs (Change). |
| equation fidelity | Exactness of placing all numerals from the problem into the correct COMPS schema slots (e.g., *Start*, *Change*, *End*; *Bigger*, *Smaller*, *Difference*). For *Combine*, part order is not penalized; for *Compare*, bigger/smaller must follow the original wording. |
| error taxonomy | Three-way categorization used in analysis: *conceptual misclassification* (wrong schema/subtype), *mapping error* (wrong numeral-to-slot assignment), and *computational error* (arithmetic mistake given a correct mapping). |
| format-compliance rate | Proportion of model outputs that conform to the required JSON schema (correct keys, types, and structure). Used to separate formatting failures from reasoning errors, though non-compliance still counts against accuracy. |

| | |
|---|---|
| generalized model equation | A template equation that captures a situation's quantitative structure (e.g., *Part + Part = Whole*). |
| graphic organizer | A visual layout (e.g., schema/diagram) used to structure problem information. |
| ground truth | reference answers or labels used to evaluate predictions. |
| hallucination | In LLMs, a fluent but incorrect or unsupported statement produced with unwarranted confidence. |
| learner model | An internal representation of a student's knowledge, skills, and misconceptions used by a tutoring system. |
| Likert scale | Ordered response scale used for attitudinal questionnaires. |
| mapping error | Correct type identified, but numerals are assigned to incorrect COMPS schema slots. |
| monorepo | a single repository containing multiple related services/components. |
| Part | A component that combines with another part to form a whole (PPW). |
| role priming | Framing a prompt by assigning the model a role (e.g., teacher, grader) to guide style and behavior. |
| scaffold | An instructional support that guides learners step by step and is gradually faded as competence increases. |
| schema mapping | The assignment of story quantities to the roles in a generalized model. |
| Smaller | The smaller of two compared quantities (Compare). |
| Start | The amount before a change occurs (Change). |
| story-grammar prompt | A guided comprehension question that helps map story details to model roles. |
| SUS–Kids | Child-friendly adaptation of the System Usability Scale for ages 7–11. |

| | |
|---|---|
| think-aloud | Protocol in which participants verbalize their thoughts while performing a task. |
| time-on-task | the time a learner spends actively working on an item or session. |
| unknown position | Which role in the model is the unknown (e.g., part vs. whole; start vs. change vs. end). |
| virtual manipulative | Interactive digital objects that help students model quantities and operations. |
| visual cueing | Use of highlights, color, or emphasis to direct attention to relevant information. |
| Whole | The total amount composed of parts (PPW). |
| word problem | A mathematics problem stated as a short story that must be interpreted and modeled to solve. |

# Licence

## Non-exclusive licence to reproduce thesis and make thesis public

I, **Chioma Jessica Nkem-Eze**,
*(*author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   **Design and Evaluation of an AI-Assisted COMPS Tutor for Students with Learning Difficulties in Mathematics**,
   *(*title of thesis)

   supervised by Eduard Barbu, PhD and Kateryna Lipmaa, PhD.
   *(*supervisor's name)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Chioma Jessica Nkem-Eze
***11/08/2025***