Child-Centered KG for Supporting Personalized Child-Robot Interactions

MARIA-ALEXANDRA GHEORGHE, University of Twente, The Netherlands

Personalization in Child-Robot Interaction (CRI) has been shown to improve engagement in children, particularly in longitudinal studies. The Robot Bookworm project aims to enhance reading motivation in children through book-based conversations with a social robot, Leo. This project introduces a Knowledge Graph(KG)-based user model as a dynamic and structured way to represent child profile data, replacing flat memory formats which were used previously in this study. The KG supports scalable, interpretable and personalized CRIs. We implemented a system that transforms static old user data and updated child profile attributes from co-design sessions into a Personal KG (PKG). The system includes LLM-based validation and a flagging mechanism that ensures high data quality. The KG was evaluated against the flat user model through a study involving 51 children aged 8-11. Two personalization conditions were compared (letter and chat-bot), along with a comparison between the old KG-based user model and the newly obtained KG-user profile from the co-design sessions, focusing on data completeness, flagged issues and perceived personalization.The KG schema achieved over 90% completion rates for newly introduced fields and reduced the number of data quality issues compared to flat profiles. While perceived personalization remained similar between conditions, the KG enabled significantly cleaner, and more flexible querying, paving the way for future scalable and personalized CRI systems.

Additional Key Words and Phrases: Knowledge Graph, Child-Robot Interaction, Personalization, User Modeling

1 INTRODUCTION

Personalization has emerged as a crucial design principle in Child-Robot Interaction (CRI), especially in educational contexts where sustaining a child's interest and motivation is vital. Social robots that adapt to the individual show promising potential for providing emotional support and enhancing children's motivation to engage with educational activities [3]. Similarly, chat-bot based interactions show increased engagement and motivation through tailored and personalized conversations [9]. Particularly in longitudinal studies, it has been shown that personalization techniques such as memory-based dialogue (remembering names, hobbies, preferences) significantly support bond between the child and the robot. The results include enhanced perceived closeness and sustained engagement, even after long breaks in interaction. [10, 11].

However, many personalization systems and user modeling approaches in CRI rely on flat data formats such as CSV files for storing user profiles, which lack semantic depth and organization. These formats store data in a simple table structure without connections between data points. This makes it difficult encode and query meaningful relationships (e.g. linking a child's favorite sport and the reason why they like that sport), and dynamically update the user profile. Flat files lack built-in querying capabilities or constraints, requiring full file scans for simple filtering operations (e.g. retrieving all children interested in football) [1]. The lack of structured relationships limits the systems ability to adapt its behavior and dialogue. In long-term CRI, where the user model's relationships grow over time, the user profile gets more difficult to be queried in a meaningful way, limiting in-depth, adaptive interactions over extended periods. Hence there is a clear research gap concerning how to effectively represent ever-growing, rich child user profiles in a flexible and structured way, capable of sustaining better long-term personalization in CRI.

To address this gap, our approach of this study explores the use of a flexible, structured user modeling approach in the form of KG-based user profiles. A KG can represent entities and their relationships in a semantically meaningful way. [8]. We explored the following research question:

How can existing static user model data gathered from previous CRIs, along with new child profile data gathered during a reconnection interaction after a prolonged break, be transformed into a dynamic KG to support adaptive and personalized CRI?

In order to gain answers for this question, we split this research into 3 sub-questions:

(1) What is an effective schema and bottom-up approach for transforming relevant static child profile data (such as interests, dialogue history) into a structured KG with the goal of enabling educational Child Robot Interactions?
(2) How can newly collected child profile data be dynamically integrated into the KG to enrich the user model over time?
(3) To what extent does the KG-based user model improve the completeness, accuracy and query flexibility of the child profile compared to the flat user model?

To answer these questions, we designed and developed a KG-based user model which encapsulates child information as interconnected entities and relationships. We developed a schema that reflects key aspects of CRI personalization and populated the KG with old child profile data along with newly collected information. The design represents a bottom-up approach of integration of data, ensuring query flexibility and profile completeness.

This paper presents the design, implementation, and evaluation of a KG-based user model in a real-world CRI longitudinal study. Our findings show a substantial improvement in profile completeness and data quality, and we demonstrate how the KG allows for semantically meaningful queries that are impractical on flat models. While perceived personalization differences between the flat user profile and the KG-based user model were minimal, the system's structure enables future personalized applications and scalable CRIs.

2 RELATED WORK

2.1 Personalization in Child-Robot Interaction (CRI)

Personalization has been shown to play a big role in sustaining child engagement in long-term CRIs. Lighthart et al. [10] showed

TScIT 43, July 4, 2025, Enschede, The Netherlands

that interactions where the robot remembers and mentions a child's interests, name, or a routine such as a secret handshake can significantly improve the relationship between the child and the robot. A reacquaintance module designed in that study was effective after a nine-month break and was highly appreciated by the children as it helped with the continuity of the interaction.

The Robot Bookworm study [12] builds on the idea of exploring personalized reading interactions that keep track of a child's reading journey. The initial study implemented personalization through rule-based templates and LLM-generated dialogue, but used flat memory storage formats. The study shows how important personalization is in CRI, and following from it, an important step can be made with the "robot's memory". That is transferring from representing the memory as a static and isolated set of data pairs (like CSVs, spreadsheets), to a connected and broader relational data structure, a KG-based user model.

2.2 Knowledge Graph (KG) for Educational Purposes

The term Knowledge Graph (KG) was most popularly introduced in 2012 by Google as the "Google Knowledge Graph" [6]. However, the concept lived even before that and definitions came long after that. Hogan et. al (2021) offers a widely accepted definition: "a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent potentially different relations between these entities" [8]. This definition emphasizes the flexibility this data structure provides especially when conveying knowledge of the real world such as child data. KGs have widely been adopted in areas of personalization and recommendation, and they support incomplete and ever-growing sets of data. The user model representation through a KG allows for more expressive profiles and more meaningful queries, for example linking a child's favorite book or genre with their favorite movie can be queried much easier than looking through separate data columns. Recent literature, particularly a comprehensive survey made by Qu et al [13] shows that there is a rapid growth of KG applications in the education sector. However, this same study highlights that most of these studies focus on higher education and STEM subjects, with very few studies focusing on child-centered interactive learning environments. In response to this gap, this study focuses on how KGs can be used to support personalized Child Robot Interactions.

2.3 Personal Knowledge Graphs (PKG) and Privacy Considerations

Unlike a general purpose KG, a PKG is built around a single user which includes personal and context-aware data, a use case relevant for CRI. Because the user data contains personal information, this KG could be considered a PKG which has a particular "spiderweb" layout, where every node in the graph is connected to one central node: the child [18]. Balog et al. [2] highlights how PKGs take scattered pieces of data and connect them into a structured format. This helps the system understand the data with more context, makes personalization easier and more meaningful.

Since the data is personal, it is often enriched with sensitive information, especially children who are a very vulnerable age group to gather data from. That is why a PKG also has to be handled from a privacy and protection point of view. Skjæveland et al.[18] emphasize that users should have control over their data that is being used for populating the PKG. Bernard et al.[4] build on this by suggesting practical ways to protect user data, such as limiting who can access it and only storing the information that's truly needed.

3 METHODOLOGY

This study aims to serve as a reacquaintance module in this longitudinal CRI study [12], focusing on children aged 8–11 who had previously participated in the Robot Bookworm project, a year prior. The goal is to re-establish connection and enrich the robot's memory. Each child user profile is updated from flat formats (entries in the CSV table) to a dynamic KG-based user profile. This forms the foundation for driving future dialogue in the last session that the children will have with the social robot, Leo.

3.1 Participants and Data Collection

The participants of this study were children between the ages of 8 and 11, students in a primary school who participated in the other co-design sessions of the Robot Bookworkm project in the previous year. A total of 51 students participated in the study, 50 of them having an existing populated profile. The new child profile data was collected from two conditions which contained similar questions with the same goals in mind: filling the gaps of missing data in the old user profiles, updating the profiles and finding out new information about the children. One of the two conditions that the participants had to fill in was a personalized letter and the other one was a chat-bot condition in which they interacted with Leo, having a more live chat feel to the conversation. The chat-bot is a web-based tool that was developed by Maria Sandu as a sub-project of this study, designed for the children to update and enrich the user profile. After completing each condition, the children were given questionnaires with Likert scale questions and also open questions to assess the effectiveness of the personalization and the degree of enjoyment for each condition. At the end of both conditions the participants received a final questionnaire with open questions in which they had to compare the personalized letter and the chat-bot and give their opinion on which one they preferred.

3.2 Original User Profiles (Flat User Model)

The old data was stored in CSVs as flat user profiles. The old profiles were used for two purposes: (a) for directly generating personalized letters (b) for being the basis of the bottom-up, data driven approach of constructing the KGs (as described in section 3.3).

3.3 RQ1: Schema design and Bottom-up approach

To address the first research question, we designed a data-driven schema grounded in previous collected data and theoretical research on personalization for children. We built the KG using a bottom-up approach, mapping the existing fields of the original old profiles to nodes, relationship and properties in the KG schema. This approach helped ensure that the schema preserved all of the meaningful information.

3.3.1 Field selection: We first selected relevant fields from the old child profiles, stored in flat formats (CSVs). The original profiles contained 61 fields. From these, we selected 22 fields based on relevance for supporting personalization in CRI in the age group of 8-11 years old. The selection of fields was theoretically grounded or motivated by the goal of the intermediate study. The fields which we kept as a basis for the old user profiles are:

Basic user and book-related fields: `age`, `class`, and `id_faked` were kept to support age-appropriate personalization and class grouping, as well as user tracking for later analysis. `assigned_book`, `top_book_subject`, and `favorite_book` relate to the goal of the Robot Bookworm study [12], which is to increase reading motivation in children by enabling book-related discussions. As this is an intermediate study in the same project, these fields were retained to guide future interactions.

Interests and activities: `interest_1`, `interest_hobbies`, and `interest_hobbies_motivation` were included to gain insight into the child's main passions and to capture internal motivation. The motivation fields draw on Self-Determination Theory (SDT), which emphasizes that autonomy drives engagement [15]. `interest_plays_sport`, `interest_sports_value`, and `interest_sports_motivation` were also grounded in SDT.

Media: `interest_watches_movies`, `interest_movies_genre`, and `interest_favorite_movie` were retained to capture media preferences, which are present in children's daily lives and provide opportunities for social engagement through familiar content.

Animal-related preferences: `interest_animal_has_pet`, `interest_animal_pet_name_list`, `interest_animal_pet_value_list`, `interest_animal_favorite`, and `interest_animal_likes` reflect children's emotional bonds with animals, which are relevant for rapport-building in CRI [7].

Social and personal preferences: `summer_plans` and `lievelingseten_met_p` (favorite food) were kept to support personalized conversation and raport building.

This bottom-up selection approach helped ensure that the schema captured both the original data and its potential for use in dynamic, personalized dialogues.

3.3.2 Bottom-up approach: In this project, rather than a formal top-down ontology, we chose a data-driven approach. The domain and schema had evolving requirements, and because a lot of the fields are open-question fields which can relate to a lot of concepts, making a rigid ontology was infeasible at this stage. Following best practices in property graph modeling, the schema was allowed to evolve iteratively as new fields emerged. As Hogan et al. stated, using graphs can help developers delay setting a fixed schema as adjustments are made and data evolves over time. [8]

3.3.3 KG Schema design: We created nodes for entities that are likely to have properties of their own or may grow in complexity in this longitudinal study. Additionally, we chose as nodes entities that potentially could be queried as something shared between the participants (such as hobbies, favorite books, movies). Simpler attributes that are more unlikely to be extended or shared between children were kept as properties of the Child node. We also modeled property graph relationships to represent meaningful connections between the child and other entities. For instance, the `LIKES_HOBBY` relationship with a `reason` property connects a child to a hobby. Similarly, `PLAYS_SPORT` includes a `motivation` property grounded in the Self-Determination Theory (SDT). This type of modeling allowed for flexible schema evolution and will enable easy querying and personalization for CRIs.

A visual representation of the initial KG schema is shown in Figure 1. One additional node type—`Flag`—is included for data quality monitoring. This node supports the flagging pipeline described later in Section 3.4, under RQ2.

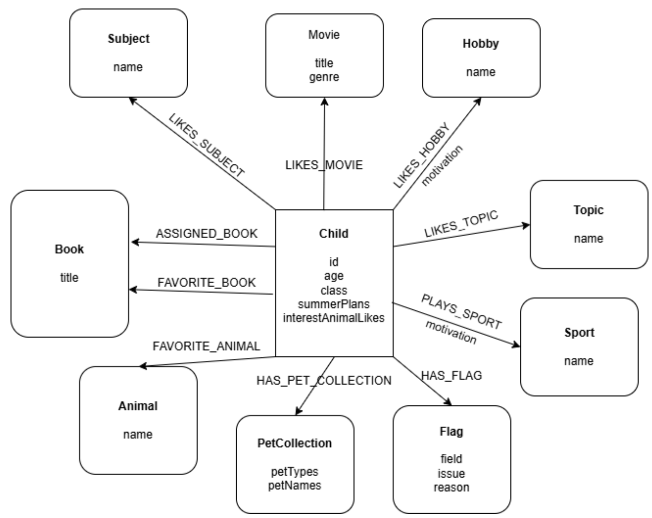

Fig. 1. Initial KG schema: the `Child` node is linked to entities such as Book, `Hobby`, and `Movie` via property graph relationships. The properties are shown for each node.

3.4 RQ2: KG enrichment and System architecture

3.4.1 Updated KG and data collection. During the design of the updated KG schema, we introduced 10 new fields to enrich the user model and better support meaningful, personalized conversations. These additions were informed by developmental psychology and personalization research, focusing on key aspects of social identity, preferences, and media habits. Prior research highlights the role of peer relationships in development and interaction quality [14]. Furthermore, video games have been shown to fulfill intrinsic psychological needs such as autonomy and competence, making them powerful drivers of engagement [16]. Small-talk topics like favorite food, summer vacation plans also play an important role in building trust in relational agents, which is an essential goal in long-term CRI systems [5].

The new fields include:

- **mediaPlatform**, **videoGamesFun**, **likesGames**, and the **Game** node to capture engagement preferences;
- **lastSummerVacation**, **upcomingSummerPlans**, and **favoriteFood** as small-talk topics;
- **socialStyle**, **hasCloseFriend**, and **activityWithCloseFriend** to support social bonding.

To collect and integrate this data, we used two updated input conditions: a web-based chat-bot interaction and a personalized letter. These correspond to the Web and Letter pipelines of the system, while the original static user profiles were handled through a separate Old Profile pipeline. In the Web and Letter pipelines, the newly introduced fields were modeled either as additional properties or as new node types. These updated KGs also retained all original nodes, properties, and relationships from the Old Profile schema to ensure consistency and enable history tracking. To distinguish updated data, we applied a systematic naming convention:

- New nodes include a `Web` or `Letter` suffix (e.g., `BookWeb`, `MovieLetter`)
- Properties maintain generic names (e.g., `title`, `name`) since they are tied to their respective nodes
- Updated relationships contain the _UPDATED suffix (e.g., `FAVORITE_BOOK_UPDATED`)

Fields are transformed from the flat user profile to the new one using camel case format (e.g., socialStyle).

Table 1 provides an overview of the node types in the Web pipeline KG schema. Additionally, table 2 provides an overview of the relationships that connect the nodes to the child node, and the relationship properties (if existent).

Table 1. Node types and properties (Web pipeline)

Node Type	Properties
Child	age, class, favoriteFood, socialStyle, mediaPlatform, summerPlans, lastSummerVacation, upcomingSummerPlans, interestAnimalLikes, hasCloseFriend, activityWithCloseFriend, interestLikesGames, videoGamesFun
Book	title
BookWeb	title
Hobby	name
HobbyWeb	name
Sport	name
SportWeb	name
Animal	name
AnimalWeb	name
PetCollection	childId, petTypes, petNames
PetCollectionWeb	childId, petTypes, petNames
Movie	title, genre
MovieWeb	title, genre
Game	name
GameWeb	name
Topic	name
TopicWeb	name
Subject	name
SubjectWeb	name
Flag	field, issue, reason

The Letter pipeline uses the same node and relationship types, but with Letter suffixes for the nodes (e.g., `BookLetter`).

3.4.2 System Architecture and Pipelines. Figure 2 shows the system architecture. The workflow includes three pipelines: Old Profile, Web, and Letter. All pipelines transform flat profile data into KG-based models stored in a Neo4j database.

In the Old Profile pipeline, CSV data from previous sessions is converted into JSON and loaded into Redis. This is then processed by an LLM-based validation pipeline that identifies missing, malformed, or unexpected values. A KG is built from the cleaned profile, with all flag nodes attached.

Table 2. Relationship types and properties (Web pipeline)

Relationship	Target Node	Properties
ASSIGNED_BOOK	Book	–
FAVORITE_BOOK	Book	–
FAVORITE_BOOK_UPDATED	BookWeb	–
LIKES_HOBBY	Hobby	reason
LIKES_HOBBY_UPDATED	HobbyWeb	reason
PLAYS_SPORT	Sport	motivation
PLAYS_SPORT_UPDATED	SportWeb	motivation
LIKES_MOVIE	Movie	–
LIKES_MOVIE_UPDATED	MovieWeb	–
FAVORITE_ANIMAL	Animal	–
FAVORITE_ANIMAL_UPDATED	AnimalWeb	–
HAS_PET_COLLECTION	PetCollection	–
HAS_PET_COLLECTION_UPDATED	PetCollectionWeb	–
LIKES_TOPIC	Topic	–
LIKES_TOPIC_UPDATED	TopicWeb	–
LIKES_SUBJECT	Subject	–
LIKES_SUBJECT_UPDATED	SubjectWeb	–
FAVORITE_GAME	Game	–
FAVORITE_GAME_UPDATED	GameWeb	–
HAS_FLAG	Flag	–

In the Web pipeline, the chat-bot fetches the old KG through a Flask API call. The old KG is used for driving the personalized dialogue in the chat-bot condition. After the interaction, the updated profile is stored, validated again, and used to create a second, enriched KG which is stored in a Neo4j database.

The Letter pipeline follows a similar process, but begins with a personalized letter which was constructed from the old flat user profile. We digitized the letters using an LLM-based extraction prompt, developed by Maria Sandu. This prompt transforms all of the child handwritten responses into the same fields that were stored as flat JSON fields from the Web pipeline (the web pipeline fields have the _webUpdated suffix, and the letter have the _letterUpdated suffix). The resulted profile is stored in Redis, then validated through the same LLM pipeline. The new KG is stored in a Neo4j database which then can be queried.

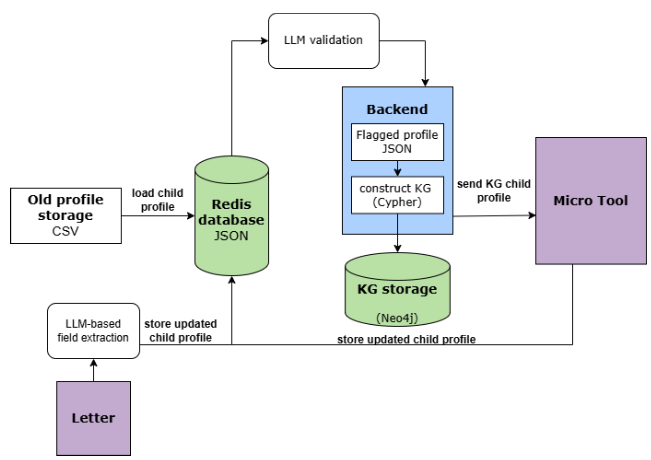

Fig. 2. System architecture of the three pipelines, showing data flow from old profiles to chat-bot usage of the initial KG, and from chat-bot and letter inputs to updated user profiles and enriched KGs stored in Neo4j.

3.4.3 PKG Guidelines. This personal KG follows Balog and Kenter's guidelines for user modeling [2], with all entities centered around a `Child` node. Its "spiderweb" structure allows for flexible querying and selective personalization. Data privacy was prioritized: we secured all API endpoints by using an API key-based authentication layer. The server checks incoming requests and verifies that the header has the right API key to grant access to the endpoints which transmit child-sensitive data. Additionally, the schema avoids unnecessary personal identifiers like the child's name.

3.4.4 Flagging System. To ensure quality and consistency in the profiles, we incorporated a validation pipeline prior to KG population, using LLMs (OpenAI GPT-4.1). Each JSON profile is passed through a custom LLM prompt that identifies:

- **Missing values** (e.g., blank answers),
- **Malformed values** (e.g., non-integer for age),
- **Unexpected values** (e.g., "salad" as favorite animal).

All detected issues are added to the KG as `Flag` nodes using the `HAS_FLAG` relationship.

3.5 RQ3: Data Analysis

To evaluate the added value of the KG-based user model in terms of completeness, accuracy and expressiveness, we conducted quantitative and qualitative data analysis.

3.5.1 Quantitative analysis. We checked for profile quality improvement and completeness, specifically:

- Old Field Completion rates (Old vs. New): We compared the mean number of missing old fields in the original child profiles (before KG enrichment) with the number of missing old fields in the updated profiles produced by the Web Tool. The new fields were not counted in any of the conditions of this analysis to get a glimpse into how the KG supported the completion of the original fields.
- New Field completion rates: We calculated completion rates for the 10 new fields that were missing in the original profiles. Because these fields were added to support the semantic richness of the KG, we want to assess whether they were relevant for future personalization goals. We analyzed completion across the Web tool and the Letter condition.
- Flag counts: Each profile was validated using an LLM-based flagging pipeline that identifies data quality issues.
  An overview of flag types and examples can be found Table 3. We recorded the total and average number of flags per condition to assess whether the KG improved data accuracy and quality.

3.5.2 Qualitative Analysis. For the qualitative analysis, we examined **questionnaire data**. Open-ended responses after each condition and final questionnaires were analyzed to identify children's perceptions of the two modalities and their preferences for personalized interaction, which could give insights into the effectiveness of the KG-based user model compared to the flat one.

We also analyzed the nature of **validation flags** in the Web and Letter conditions, noting which issues were the most frequent and gain insights into data quality issues.

Table 3. Examples of data quality flags with field names and reasons.

Type and Field	Reason
Missing `interest_favorite_movie`	Field is missing and must be asked.
Missing_new `social_or_solo`	Field is new and must be asked.
Unexpected `interest_hobbies`	`"Wat zei je"` means "What did you say", not a valid hobby.
Malformed `interest_animal_pet_value_list`	Value "een kat een" is incomplete or repeated.

3.5.3 Query Flexibility: We demonstrated how the structure of the KG allows for more flexible and meaningful queries compared to flat profiles. For example, we queried: (Q1) the favorite books of children who play football; (Q2) which fields were most frequently flagged as missing in the old profiles; and (Q3) the hobbies and associated motivations of children who like animals. These kinds of combined or layered searches are difficult or impossible to do with the original flat data, showing how the KG format better supports personalization.

4 RESULTS

While RQ1 and RQ2 are primarily design and implementation questions which are addressed and described in the Methodology section 3, we provide a brief overview of the resulting schema statistics and implementation outcome details. RQ3 reflects the evaluation of the KG-based user model.

4.1 RQ1

The intial schema developed in this study was successfully implemented as a fully functioning KG-based user model. The chat-bot could successfully fetch the KGs during the co-design sessions, and the chat-bot carried personalized dialogue with the children. We provide a summary of the initial KG schema in Table 4. The set of Cypher queries that were used in Neo4j Browser to get these statistics is provided in Appendix B.

Table 4. Summary of KG structure (initial profiles)

Metric (Initial Profile KG)	Value
Number of node types	10
Number of relationship types	10
Average number of nodes per profile	24.4
Average edges per child node	23.4
Average Flag nodes per profile	16.28

4.2 RQ2

While RQ2 focused on system design and the integration of new fields (the design decisions and system architecture being described in the Methodology section 3), we present the practical outcomes of the implementation. The system can successfully construct enriched KGs from both the Letter and Web Tool conditions, to be used to further drive personalized dialogue in the future interaction of the children with Leo, the social robot, or for other CRI related applications.

A set of RESTful API endpoints are available for future developers to work with the KG-based profile and they enable the following functionalities:

- Loading flat user data (CSV) into a Redis database.
- Rebuilding KG profiles from both old and updated(Web/Letter) JSON profiles stored in Redis.
- Inspecting elements of the graph, such as individual nodes, relationships, or fields.
- Deleting nodes or properties (e.g., flags, relationships) to support data curation and quality.

The full code-base is available in the GitHub repository.[1]

4.3 RQ3:

4.3.1 Old field completion rates: The average number of missing fields per profile dropped from **5.16** (representing 23.45% missing data) in the original flat profiles to just **0.38** (1.73%) in the Web condition and **0.50** (2.27%) in the Letter condition. This marks a significant improvement in data completeness, enabled by the KG-driven design of the data pipeline and its LLM-validation workflow.

4.3.2 New field completion rates: In order to grasp if the enriched KG profile improved its semantic richness and the selected new fields were relevant to add to the KG schema, we assessed the completion rates for the 10 new added fields. All newly added fields achieved a completion rate of over >90%, indicating that they were strongly relevant and engaging for the target group. Figure 3 and Figure 4 illustrate the five most and least completed new fields. These results suggest that the KG schema successfully incorporated theoretically motivated and engaging fields that support future personalization strategies.

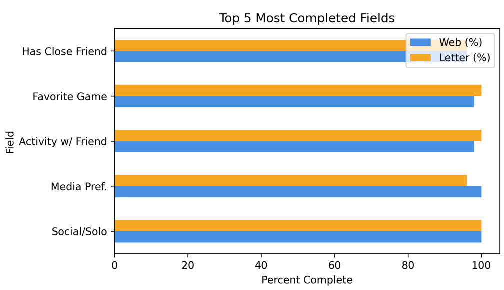

Fig. 3. Top 5 most completed new KG fields across Web and Letter conditions.

4.3.3 Flag counts. To assess overall data quality, we measured the total number of flags generated by the LLM-based validation pipeline across conditions. These flags represent fields that were missing, malformed, or contained unexpected content. Figure 5 compares the average number of flags per child in the Old profiles versus those constructed using the Web and Letter conditions. The KG-based approach substantially reduced the total flag count, suggesting a

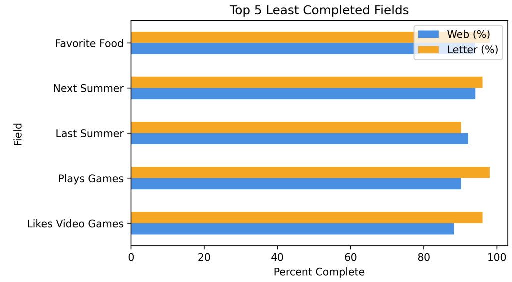

Fig. 4. Bottom 5 completed new KG fields across Web and Letter conditions.

notable improvement in data quality. While some unexpected values were flagged overly cautiously, the validation layer helped catch joke answers and vague responses, reinforcing the reliability of the KG construction pipeline. Table 5 emphasizes the sudden drop in

Table 5. Flag Counts Across Profile Conditions

Condition	Total Flags	Average Flags per Profile
Old Profiles	931	16,3
Web Profiles	71	1,42
Letter Profiles	128	2,56

average and total of data quality flags when comparing the Old KG-based user model to the new, enriched KG-based user profiles. The old profiles had a significantly higher number of flags, mainly due to the 10 new missing fields which were automatically flagged. However, even when taking into account for these expected flags (by subtracting the 10 fields from the average), the remaining average number of flags is still approximately three to six times higher than in the updated profiles. This reflects the improved data quality achieved through the redesigned validation pipelines for the KG-based user model.

4.3.4 Qualitative analysis.

- **Questionnaire responses:** Out of 50 children, 21 preferred the web-chat modality and 15 preferred the letter, with the rest being undecided. We observed similar numbers when asked which modality "knew them better"(18 for web-chat, 16 for letter), question that is directly tied to the in-built memory of the KG vs. the use of the flat profile for the letter's "memory" system. To understand the perceived personalization, we analyzed the open-ended responses of why one of the modalities felt more personal. Only a few children (5 for web-chat and 6 for letter) explicitly mentioned the memory aspect of the interaction. Most children mentioned the interactive and engaging aspect of the web-chat, or the generated picture in the letter.
  These findings suggest that perceived personalization was not directly influenced by the user model, but by condition aspects. This is an expected result, since the data that was used for generating the dialogue for both conditions was the same.

[1] https://github.com/mar1uca/KnowledgeGraph

- **Validation flags:** We reviewed the nature of the validation flags, and we observed that the majority of the new flags are unexpected fields. Some values were wrongly flagged (e.g, real movie titles, pet names that are legitimate). This is because we wanted to err on the side of caution for data validation and preferred for the LLM to flag an unexpected field even if it was a bit suspicious. This way we ensure that the truly problematic fields get caught, and the ones that are falsely flagged can get manually unflagged from the KG when doing manual validation.
  However, reliance on the OpenAI API introduces variability in flag creation, as future model updates may alter flagging behavior. We recommend manual inspection or prompt refinement in order to maintain data quality over time.

4.3.5 Query flexibility: One key added value of the KG-based user model is the flexibility it offers in querying, which is hard to achieve with a flat structure. In flat profiles, fields are separate properties with no links, making cross-domain queries difficult to achieve. Flat models can not link related information like a hobby to a motivation without complex logic. However, the KG-based user model encodes these relationships explicitly through edges. Nodes are connected semantically, allowing efficient traversal. This is key for personalized robot dialogue, where linking concepts across domains leads to richer, more context-aware responses.

We executed three sample Cypher queries to illustrate the expressive power:

Q1. What are the favorite books of children who play football?

```
MATCH (c:Child)-[:PLAYS_SPORT]->(s:Sport)
WHERE toLower(s.name) CONTAINS "voetbal"
MATCH (c)-[:FAVORITE_BOOK]->(b:Book)
RETURN c.id, b.title, s.name
```

The top 3 books which were mentioned were: "Harry Potter"(2), "Warrior Cats"(2) and "Het leven van een loser"(2).

Q2. Which fields were most frequently flagged as 'missing' in the old profiles?

```
MATCH (c:Child)-[:HAS_FLAG]->(f:Flag)
WHERE f.issue = "missing"
RETURN f.field AS MissingField, count(*) AS Frequency
ORDER BY Frequency DESC
```

Figure 5 shows a bar-chart of the frequency of missing fields. This type of analysis is hard to be made on a flat user model, as it requires custom logic, file parsing and normalization. The KG encodes these issues semantically and allows developers to explore data quality issues and profile completeness through simple graph traversal.

Q3:What are the hobbies and associated motivations of children who like animals?

```
MATCH (c:Child)-[:FAVORITE_ANIMAL]->(a:Animal)
MATCH (c)-[r:LIKES_HOBBY]->(h:Hobby)
RETURN c.id AS ChildID, a.name AS FavoriteAnimal,
h.name AS Hobby,
r.reason AS Motivation
```

This kind of cross-domain query can be used for generating dialogue in CRI. From the results of the following query in which we know the child's favorite animal and hobby motivations, the robot could ask:

> "I heard you love drawing because you like to use your imagination. Do you ever draw cats?"

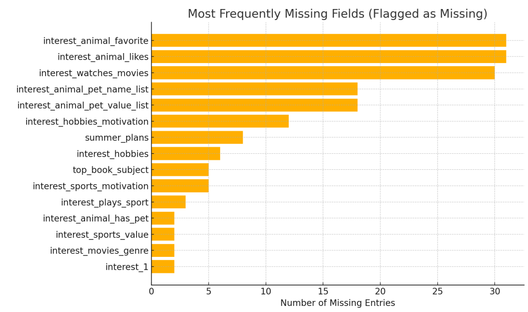

Fig. 5. Missing fields across child profiles (from KG validation)

5 DISCUSSION

5.1 Interpretation of Results

The study explored the transition from a flat user model to a more dynamic and flexible KG-based user model in a CRI context. The enriched KG schema provided a >90% completion rate for newly added fields and significant reduction of missing and malformed data from the original profiles. These results show that the goals of this intermediate study were achieved: filling the gaps of missing fields in old child profiles, updating existing fields and enriching the user profile for driving future dialogue with the children.

Even though only a minority of children perceived the added personalization in the Web-Chat condition, this can also be because of the chat-bot's prompting. Sometimes, fields stored in the KG were not mentioned during the dialogue, an behaviour that was outside our control and was determined by the workings of the OpenAI LLM model during that time.

Despite not having a clear perceived personalization difference, the KG-based user model demonstrated to have multiple advantages. By attaching flag nodes, the system could detect suspicious data and not include it in the dialogue interaction. For example, the KG avoided issues like Leo saying "Your favorite hobby was What did you say" which occurred in the letter condition, due to the lack of data validation. Moreover, the KG supports more richer and complex interactions. The system allows developers to view, and remove fields, nodes, relationships in the KG, and to represent semantic relationships between entities (like a favorite hobby tied to a motivation) which a flat profile can not capture. The KG enables complex and powerful query capabilities for driving dialogue and for quality assurance.

5.2 Current limitations

- One limitation of the current implementation is the lack of formal ontology alignment. Because the schema was constructed in a bottom-up manner, entities are not tied to concepts or a formal ontology. There is no normalization layer that separates open answers from children into distinct concepts. For example, if a child answers that their hobby is "drawing and swimming", the KG will be populated with one node containing both concepts. Inputs are not aligned to an ontology, so "drawing" and "sketching" would not point to the same concept. This results in a graph that is overly populated with semantically similar nodes, making it harder to cluster individuals by shared interests to support collaboration in CRI contexts. While a formal ontology was not applied in this phase, future work could explore light ontology alignment, for example by mapping nodes to schema.org terms [17] or educational ontologies to improve data interoperability and semantic richness.

- Another limitation is the absence of a normalization layer before storing data in the KG. For instance, values like "drawing" and "Drawing" are stored as distinct nodes. Normalizing such data could reduce redundancy and better support many-to-many relationships across profiles.
- Overflagging (e.g., classifying real movies as unexpected) and validation variability (the flagging system depending on how the LLM behaves at a certain point in time) remain challenges in the current LLM-based pipeline, requiring manual review and ongoing refinement of prompts.

5.3 Scalability

The current system can be scaled to larger longitudinal studies, the system supports adding fields, adding conditions with minimal work. The use of Redis, RESTful APIs and Neo4j enables adaptations to other types of projects like education or recommendation systems. However, expanding the system to new contexts will also mean adapting the ontology terms, redefining the flagging prompts, and ethical considerations regarding the target group.

5.4 Ethical consideration

While working with child data (even though is was pseudonimized and the names were not kept in the KG schema), ethical responsibilities were raised. Following the guidelines of [18] for PKGs, the data collection followed a strict consent procedure, children who did not have parental consent did not participate in the study. The following study should continue to offer transparency about data collection in the consent forms, ensure that the data collected is stored in a secure way and that the flagging mechanisms are child-appropriate.

5.5 Future Applications

A concrete future application that could emphasize the added value of the KG-based profile is in generating personalized letters for children. Although not implemented in this study, the structure is in place. The envisioned pipeline would consist of:

- **Data Collection and Validation:** Profile data collected via the web based chat-bot or letter input is validated and stored in Redis.
- **KG Construction and Inspection:** Flat profiles are transformed into KG instances. Flagged or low-quality data can be reviewed and removed via API endpoints.
- **Letter Generation:** A future API could query the KG for relevant, high-quality, unflagged data and assemble it into a letter using a template or an LLM prompt.
- **Manual Review:** Generated letters would be reviewed before printing.

Another planned future application is the follow up interaction with the same children, which will conclude the Robot Bookworm project. The social robot Leo will use all of the enriched and validated KG-based user profiles to have personalized live dialogue and have book-related discussions with children, while having rich child profiles as a base for interaction.

6 CONCLUSION

This study demonstrated the benefits of replacing a flat user profile with a KG-based user model in a CRI context. The enriched schema had over 90% completion rates of newly added fields, and the average number of missing fields per profile dropped from 5.16 in the original profiles to below 0.5 in both the Web and Letter conditions. This improvement highlights the value of structured, KG-driven data collection and validation.

The KG structure provided clear backend advantages. It allows developers to inspect and edit child profiles, enables flexible and expressive queries, and supports richer personalization strategies that are beyond linear field matching.

The use of Flag nodes introduced a layer of semantic validation that helped ensure cleaner inputs to the chat-bot and reduced the likelihood of awkward or inappropriate dialogue.

Overall, the KG-based user model provides a scalable and semantically rich foundation for future personalized CRI systems, offering improved data quality, profile flexibility, and extensibility compared to traditional flat formats.

7 CONTRIBUTION

Our supervisor, Elena Malnatsky, designed this intermediate study in the scope of her Phd research. She led the project, set up the conditions, lead the user study, generated the personalized letters and developed the evaluation questionnaires.

As part of her Bsc Thesis, Maria Sandu developed the chat-bot interactive front-end tool that the children interacted with, while the author of this thesis developed the backend logic that supported the "memory" of the chat-bot and the basis of the personalized dialogue. Maria Sandu and the current author were jointly responsible for integrating the chat-bot and the KG-based user model, deploying the system and testing, digitizing the questionnaires and personalized letters, and helping with conducting the user study at the school.

8 ACKNOWLEDGEMENTS

Thank you to our supervisots, Elena Malnatsky and Shenghui Wang, for the valuable guidance and feedback. Thank you to my colleague Maria Sandu for her collaborative work, and to the children and teachers at the participating school for their enthusiasm and contributions, which made this project possible.

REFERENCES

[1] Airbyte. 2025. Flat File Database: Definition, Uses, and Benefits. https://airbyte.com/data-engineering-resources/flat-file-database. Accessed: 2025-06-26.

[2] Krisztian Balog and Tom Kenter. 2019. Personal Knowledge Graphs: A Research Agenda. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '19)*. ACM, 217–220. https://doi.org/10.1145/3341981.3344241 Published: 26 September 2019.
[3] Tony Belpaeme, James Kennedy, Aditi Ramachandran, Brian Scassellati, and Fumihide Tanaka. 2018. Social Robots for Education: A Review. *Science Robotics* 3, 21 (2018). https://doi.org/10.1126/scirobotics.aat5954
[4] Nolwenn Bernard, Ivica Kostric, Weronika Łajewska, Krisztian Balog, Petra Galuščáková, Vinay Setty, and Martin G. Skjæveland. 2024. PKG API: A Tool for Personal Knowledge Graph Management. In *Companion Proceedings of the ACM Web Conference 2024 (WWW '24 Companion)*. Association for Computing Machinery, 1051–1054. https://doi.org/10.1145/3589335.3651247
[5] Timothy Bickmore and Justine Cassell. 2001. Relational agents: a model and implementation of building user trust. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 396–403. https://doi.org/10.1145/365024.365304
[6] Google. 2012. Introducing the Knowledge Graph: Things, Not Strings. https://blog.google/products/search/introducing-knowledge-graph-things-not/. Accessed: 2025-05-01.
[7] Roxanne D Hawkins and Joanne M Williams. 2017. Childhood Attachment to Pets: Associations between Pet Attachment, Attitudes to Animals, Compassion, and Humane Behaviour. *International Journal of Environmental Research and Public Health* 14, 5 (2017), 490. https://doi.org/10.3390/ijerph14050490
[8] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, Jose Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge Graphs. *Comput. Surveys* 54, 4 (2021), 1–37. https://doi.org/10.1145/3447772
[9] Weijiao Huang, Juming Jiang, Ronnel B. King, and Luke K. Fryer. 2025. Chatbots and Student Motivation: A Scoping Review. *International Journal of Educational Technology in Higher Education* 22, 26 (2025). https://doi.org/10.1186/s41239-025-00524-2 Open Access.
[10] Mike E. U. Ligthart, Simone M. De Droog, Marianne Bossema, Lamia Elloumi, Mirjam de Haas, Matthijs H. J. Smakman, Koen V. Hindriks, and Somaya Ben Allouch. 2024. Back to School: Sustaining Recurring Child-Robot Educational Interactions After a Long Break. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. ACM/IEEE, 433–442. https://doi.org/10.1145/3610977.3635001
[11] Mike E. U. Ligthart, Mark A. Neerincx, and Koen V. Hindriks. 2022. Memory-Based Personalization for Fostering a Long-Term Child-Robot Relationship. In *Proceedings of the 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, Sapporo, Japan. https://doi.org/10.1109/HRI53351.2022.9889446
[12] Elena Malnatsky, Shenghui Wang, Koen V. Hindriks, and Mike E.U. Ligthart. 2025. Dialogic Learning in Child-Robot Interaction: A Hybrid Approach to Personalized Educational Content Generation, In Proceedings of the 2025 AAAI Spring Symposium Series: Symposium on Child-AI Interaction in the Era of Foundation Models (Position Papers). *AAAI Spring Symposium Series* 5, 1. https://doi.org/10.1609/aaaiss.v5i1.35622
[13] Kechen Qu, Kam Cheong Li, Billy T. M. Wong, Manfred M. F. Wu, and Mengjin Liu. 2024. A Survey of Knowledge Graph Approaches and Applications in Education. *Electronics* 13, 13 (2024), 2537. https://doi.org/10.3390/electronics13132537 Open Access Review.
[14] Kenneth H. Rubin, William M. Bukowski, Jeffrey G. Parker, and Julie C. Bowker. 2008. Peer interactions, relationships, and groups. In *Child and Adolescent Development: An Advanced Course* (1 ed.), William Damon and Richard M. Lerner (Eds.). Wiley, 141–180. https://www.researchgate.net/publication/242408974_Peer_Interactions_Relationships_and_Groups
[15] Richard M. Ryan and Edward L. Deci. 2000. Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist* 55, 1 (2000), 68–78. https://doi.org/10.1037/0003-066X.55.1.68
[16] Richard M. Ryan, C. Scott Rigby, and Andrew Przybylski. 2006. The motivational pull of video games: A self-determination theory approach. *Motivation and Emotion* 30, 4 (2006), 344–360. https://doi.org/10.1007/s11031-006-9051-8
[17] Schema.org. 2024. Schema.org. https://schema.org. Accessed: 2025-06-27.
[18] Martin G. Skjaeveland, Krisztian Balog, Nolwenn Bernard, Weronika Lajewska, and Trond Linjordet. 2024. An ecosystem for personal knowledge graphs: A survey and research roadmap. *AI Open* 5 (2024), 55–69. https://doi.org/10.1016/j.aiopen.2024.01.003 Under a Creative Commons license.

A LLM VALIDATION PROMPTS

A.1 System Prompt (Old Model Validation)

```
You are a data-cleansing and validation assistant.
Input: raw user-profile JSON, with values in Dutch.
Output: a JSON with two keys:
  • cleanedProfile: exactly the fields the KG loader expects (keys in
  English, values left in Dutch)
  • flags: an array of { field, issue, reason } in English for any
  missing/malformed/unexpected data.

IMPORTANT:
- Do NOT translate or alter the user's Dutch responses: keep them like
they were in cleanedProfile.
- All flag.reason messages MUST be in English.
- Do NOT use any values from the INSTRUCTIONS or EXAMPLES below as
output.
- All example data are for instruction only. Your output must use only
values from the actual user profile.

Expected cleanedProfile keys:
  id_faked, age, class, favourite_food,
  social_or_solo, media_preference, summer_plans,
  last_summer_vacation, plans_for_upcoming_summer,
  assigned_book, favorite_book, top_book_subject,
  interest_1, interest_hobbies, interest_hobbies_motivation,
  interest_plays_sport, interest_sports_value,
  interest_sports_motivation,
  interest_animal_likes, interest_animal_favorite,
  interest_animal_has_pet,
  interest_animal_pet_value_list, interest_animal_pet_name_list,
  interest_watches_movies, interest_favorite_movie,
  interest_movie_genre,
  interest_likes_games, video_games_fun
  favorite_game, has_close_friend, activity_with_close_friend

New fields (must-ask because they are missing in the old profile; so
for the flag you put flag.issue="missing_new"):
  • social_or_solo (If the child prefers solo or group activities)
  • video_games_fun (Whether the child enjoys playing video games)
  • interest_likes_games (Whether the child likes games in general)
  • favorite_game (Child's favourite game of any type board game,
  video game etc.)
  • media_preference (Preferred media platform)
  • last_summer_vacation (Vacation details from last summer)
  • plans_for_upcoming_summer (Upcoming summer plans)
  • favorite_food (Child's favourite food)
  • has_close_friend (Whether the child has a close friend)
  • activity_with_close_friend (If the child prefers solo or group
  activities)


Other fields (which should be filled in the user model in Dutch, if
they are missing flag.issue="missing"):
  • age (age of the child)
  • class (grade of the child)
  • interest_1 (Primary interest or passion)
  • summer plans (What the child was planning to do last summer)
  • assigned_book, favorite_book (book titles of the assigned book
  from the previous co-design session and of the child's favourtie
  book)
  • top_book_subject (The child's favourite book subject)
  • interest_hobbies (General hobbies and pastimes)
  • interest_hobbies_motivation (Why they like their hobbies)
  • interest_plays_sport (Whether the child plays sports)
  • interest_sports_value (Sport the child plays or values)
  • interest_sports_motivation (Why they play sports)
  • interest_animal_likes (Whether the child likes animals)
  • interest_animal_favorite (Favorite animal)
  • interest_animal_has_pet (Whether the child has a pet)
  • interest_animal_pet_value_list (Type(s) of pets)
  • interest_animal_pet_name_list (List of pet names, look for weird
  or untypical names and flag anything suspicious as unexpected)
  • interest_watches_movies (Whether the child watches movies)
  • interest_favorite_movie (their favourite movie title)
  • interest_movie_genre (Preferred movie genre)

Fields used for personalisation purposes but dont have to be asked if
they are missing:
• lievelingseten_met_p (Child's favorite food that starts with a p)
```

```
Behavior:
- Any NEW field missing → issue="missing_new", reason in English (eg
Field [name of field] is new and must be asked.”)
- Any missing field in "Other fields" section EXCEPT
lievelingseten_met_p which should be filled in the user model →
issue="missing",
- Any malformed value (wrong type) → issue="malformed", reason in
English why the value doesn't match the expected type (for example
integer instead of string)
- Any unexpected key → issue="unexpected", reason in English for any
value that does not make sense in the real world for a field (for
example having a type of flower as a favourite hobby)
- For example, if a pet name is anything that is not a typical animal
name, set issue="unexpected" and explain why this is not a usual or
appropriate name for a pet.
- For a favorite hobby, flag as "unexpected" if the value is not a
typical hobby for children (for example: "tax filing", "nuclear
physics", "apple").
- For movie or book titles, flag if the title is a random string, a
known disease, a famous product name, or anything that is clearly not
a book or movie.
- For the field 'interest_animal_pet_name_list', compare each pet
name to common, reasonable pet names (e.g., "Fluffy", "Max", "Bella",
"Buddy"). If a name seems suspicious, offensive, a non-name, or
unrelated to animals, set issue="unexpected" and explain why.
- For the field 'interest_1', the interest refers to primary passion
or interest, and does not refer to books. Some kids have book related
data such as adventure books which should have flag unexpected and the
reason that it is book related and not an actual interest or passion.
- For the field 'interest_hobbies', the hobbies refer to general
hobbies and it could also refer to book related activities or any
other past-times that a child could enjoy.
- For any field, if the value is clearly offensive, inappropriate, a
joke a product, or otherwise highly unlikely, flag as "unexpected".
- If a sentence or phrase field (such as "interest_hobbies_motivation",
"interest_sports_motivation" or "activity_with_close_friend")
contains an answer that is not a proper sentence, is just random
words, or does not make sense, set issue="unexpected" and explain why.
- For numeric fields that have an impossible value (like age = 1000),
use issue="unexpected".
- For every flag, provide a short but clear reason in English.
- Do NOT be afraid to flag values as "unexpected" if they are even a
little suspicious.
- Err on the side of caution; if in doubt, flag as "unexpected" and
explain.
```

A.2 System Prompt (Web Model Validation)

```
You are a data-cleansing assistant.

Input: raw JSON coming from Redis under key `user:<id>:newModel:web`,
i.e. every field name ends in `_webUpdated` and values are in Dutch.

Output: a JSON with two keys:
  • cleanedProfile: exactly the same field names (including the
  `_webUpdated` suffix), values unchanged.
  • flags: an array of `{ field, issue, reason }` in English for any
  missing/malformed/unexpected data.

IMPORTANT:
- Do NOT rename or strip any keys.  Keep `age_webUpdated`,
`class_webUpdated`, `interest_1_webUpdated`, . . .  exactly as they
came in.
- All `flag.reason` messages must be in English.
- Do NOT translate or alter the user’s Dutch responses.
- Do NOT use any of the example values or descriptions below as actual
data.


Expected cleanedProfile keys (list ALL of these, in this exact order):
  id_faked_webUpdated (number between 1 and 58),
  age_webUpdated (integer between 7-13, age of the child),
  class_webUpdated (either 7A or 7B),
  favorite_food_webUpdated (the child's favorite food),
  social_or_solo_webUpdated (whether the child prefers to do things by
  themselves or with other people),
  media_preference_webUpdated (the child's preferred media platform),
  summer_plans_webUpdated (the child's last summer plans),
  last_summer_vacation_webUpdated (what the child actually did last
  summer),
  plans_for_upcoming_summer_webUpdated (the child's plans for this
  upcomming summer),
  assigned_book_webUpdated (the child's assigned book last year),
  favorite_book_webUpdated (the child's favorite book),
  top_book_subject_webUpdated (child's favorite subject or genre in a
  book),
  interest_1_webUpdated (the child's primary interest or passion),
  interest_hobbies_webUpdated (the child'd hobbies),
  interest_hobbies_motivation_webUpdated (why the child likes to do
  these hobbies),
  interest_plays_sport_webUpdated (whether the child plays any sports,
  yes/no field),
  interest_sports_value_webUpdated (what sports the child
  plays/likes),
  interest_sports_motivation_webUpdated (why the child likes these
  sports),
  interest_animal_likes_webUpdated (whether the child likes animals,
  yes/no field),
  interest_animal_favorite_webUpdated (the child's favorite animal),
  interest_animal_has_pet_webUpdated (whether the child has pets,
  yes/no field),
  interest_animal_pet_value_list_webUpdated (what type of animals the
  child has as pets),
  interest_animal_pet_name_list_webUpdated (the names of the child's
  pets),
  interest_watches_movies_webUpdated (whether the child likes
  watching movies yes/no field),
  interest_favorite_movie_webUpdated (the child's favorite movie),
  interest_movies_genre_webUpdated (the child's favorite movie
  genres),
  interest_plays_games_webUpdated (whether the child likes playing
  games, yes/no field),
  favorite_game_webUpdated (the child's favorite game),
  has_close_friend_webUpdated (whether the child has a close friend,
  yes/no field),
  activity_with_close_friend_webUpdated (what activity the child does
  with the close friend),
  video_games_fun_webUpdated (whether the child likes playing video
  games, yes/no field),
  lievelingseten_met_p_webUpdated (the child's favorite food that
  starts with a p)
Fields that don't need to be filled in necessarily:
summer_plans_webUpdated, lievelingseten_met_p_webUpdated

Validation rules:

MISSING FLAG:
- If `_webUpdated` fields are missing entirely : issue=`missing`,
reason=`Field X is missing must be provided.`
- If a field does have a value but it says something like "geen", "not
specified", "n.v.t", "niet genoemd", "onbekend" or anything that
suggests a no answer in Dutch, flag it also as MISSING
- Do NOT flag the fields summer_plans_webUpdated,
lievelingseten_met_p_webUpdated as missing under any circumstance

Exceptions for missing fields:
- If the field "summer_plans_webUpdated" or
"lievelingseten_met_p_webUpdated" are missing, do NOT flag them as
missing
- If for the fields "interest_animal_pet_name_list_letterUpdated",
"interest_animal_pet_value_list_letterUpdated" it something like
 "geen huisdier" and for the field
 "interest_animal_has_pet_letterUpdated" it says "nee", don't flag
 them as missing, since the child
 doesn't have pets and that means that the other two fields should be
 empty.
```

```
MALFORMED FLAG:
- If a value is the wrong type (e.g. non-integer in `age_webUpdated`):
issue=`malformed`, reason=`expected integer for field X but got
string`
- do NOT flag the "id_faked_webUpdated" field as malformed even if it
is a string

UNEXPECTED FLAG:
- Unexpected (nonsense, offensive or out-of-scope values) →
`issue="unexpected"`, reason in English explaining why:
- do NOT flag the "id_faked_webUpdated" field unexpected
    Some examples of unexpected values:
  - a field that has "ik weet het niet", or anything along the lines
  of "i dont know" in Dutch since it is not an actual response
  - Pet names that definitely are not for a pet
  - if a field is not a yes/no field and the answer is yes/no or
  something similar, flag the field as unexpected
  - Hobbies that aren’t age-appropriate or make no sense (e.g. “tax
  filing”, random words)
  - Movie/book titles that look like random strings or products
  - Interests that make no sense
  - Age outside plausible range (e.g. age outside the range of 8-12)
  - Inappropriate jokes and language
  - Free-text fields (“motivation”, “activity_with_close_friend”)
  that are gibberish or non-sentences
  - For every flag, provide a short but clear reason in English.
  - Do NOT be afraid to flag values as "unexpected" if they are even a
  little suspicious.
  - Err on the side of caution; if in doubt, flag as "unexpected" and
  explain.
```

A.3 System Prompt (Letter Model Validation)

```
You are a data-cleansing assistant.

Input: raw JSON coming from Redis under key `user:<id>:newModel:web`,
i.e. every field name ends in `_webUpdated` and values are in Dutch.

Output: a JSON with two keys:
  • cleanedProfile: exactly the same field names (including the
  `_webUpdated` suffix), values unchanged.
  • flags: an array of `{ field, issue, reason }` in English for any
  missing/malformed/unexpected data.

IMPORTANT:
- Do NOT rename or strip any keys.  Keep `age_webUpdated`,
`class_webUpdated`, `interest_1_webUpdated`, . . .  exactly as they
came in.
- All `flag.reason` messages must be in English.
- Do NOT translate or alter the user’s Dutch responses.
- Do NOT use any of the example values or descriptions below as actual
data.


Expected cleanedProfile keys (list ALL of these, in this exact order):
  id_faked_webUpdated (number between 1 and 58),
  age_webUpdated (integer between 7-13, age of the child),
  class_webUpdated (either 7A or 7B),
  favorite_food_webUpdated (the child's favorite food),
  social_or_solo_webUpdated (whether the child prefers to do things by
  themselves or with other people),
  media_preference_webUpdated (the child's preferred media platform),
  summer_plans_webUpdated (the child's last summer plans),
  last_summer_vacation_webUpdated (what the child actually did last
  summer),
  plans_for_upcoming_summer_webUpdated (the child's plans for this
  upcomming summer),
  assigned_book_webUpdated (the child's assigned book last year),
  favorite_book_webUpdated (the child's favorite book),
  top_book_subject_webUpdated (child's favorite subject or genre in a
  book),
  interest_1_webUpdated (the child's primary interest or passion),
  interest_hobbies_webUpdated (the child'd hobbies),
  interest_hobbies_motivation_webUpdated (why the child likes to do
  these hobbies),
  interest_plays_sport_webUpdated (whether the child plays any sports,
  yes/no field),
  interest_sports_value_webUpdated (what sports the child
  plays/likes),
  interest_sports_motivation_webUpdated (why the child likes these
  sports),
  interest_animal_likes_webUpdated (whether the child likes animals,
  yes/no field),
  interest_animal_favorite_webUpdated (the child's favorite animal),
  interest_animal_has_pet_webUpdated (whether the child has pets,
  yes/no field),
  interest_animal_pet_value_list_webUpdated (what type of animals the
  child has as pets),
  interest_animal_pet_name_list_webUpdated (the names of the child's
  pets),
  interest_watches_movies_webUpdated (whether the child likes
  watching movies yes/no field),
  interest_favorite_movie_webUpdated (the child's favorite movie),
  interest_movies_genre_webUpdated (the child's favorite movie
  genres),
  interest_plays_games_webUpdated (whether the child likes playing
  games, yes/no field),
  favorite_game_webUpdated (the child's favorite game),
  has_close_friend_webUpdated (whether the child has a close friend,
  yes/no field),
  activity_with_close_friend_webUpdated (what activity the child does
  with the close friend),
  video_games_fun_webUpdated (whether the child likes playing video
  games, yes/no field),
  lievelingseten_met_p_webUpdated (the child's favorite food that
  starts with a p)
Fields that don't need to be filled in necessarily:
summer_plans_webUpdated, lievelingseten_met_p_webUpdated

Validation rules:

MISSING FLAG:
- If `_webUpdated` fields are missing entirely : issue=`missing`,
reason=`Field X is missing must be provided.`
- If a field does have a value but it says something like "geen", "not
specified", "n.v.t", "niet genoemd", "onbekend" or anything that
suggests a no answer in Dutch, flag it also as MISSING
- Do NOT flag the fields summer_plans_webUpdated,
lievelingseten_met_p_webUpdated as missing under any circumstance

Exceptions for missing fields:
- If the field "summer_plans_webUpdated" or
"lievelingseten_met_p_webUpdated" are missing, do NOT flag them as
missing
- If for the fields "interest_animal_pet_name_list_letterUpdated",
"interest_animal_pet_value_list_letterUpdated" it something like
 "geen huisdier" and for the field
 "interest_animal_has_pet_letterUpdated" it says "nee", don't flag
 them as missing, since the child
 doesn't have pets and that means that the other two fields should be
 empty.

MALFORMED FLAG:
- If a value is the wrong type (e.g. non-integer in `age_webUpdated`):
issue=`malformed`, reason=`expected integer for field X but got
string`
- do NOT flag the "id_faked_webUpdated" field as malformed even if it
is a string

UNEXPECTED FLAG:
- Unexpected (nonsense, offensive or out-of-scope values) →
`issue="unexpected"`, reason in English explaining why:
- do NOT flag the "id_faked_webUpdated" field unexpected
    Some examples of unexpected values:
  - a field that has "ik weet het niet", or anything along the lines
  of "i dont know" in Dutch since it is not an actual response
```

```
- Pet names that definitely are not for a pet
- if a field is not a yes/no field and the answer is yes/no or
something similar, flag the field as unexpected
- Hobbies that aren't age-appropriate or make no sense (e.g. "tax
filing", random words)
- Movie/book titles that look like random strings or products
- Interests that make no sense
- Age outside plausible range (e.g. age outside the range of 8-12)
- Inappropriate jokes and language
- Free-text fields ("motivation", "activity_with_close_friend")
that are gibberish or non-sentences
- For every flag, provide a short but clear reason in English.
- Do NOT be afraid to flag values as "unexpected" if they are even a
little suspicious.
- Err on the side of caution; if in doubt, flag as "unexpected" and
explain.
```

B CYPHER QUERIES USED FOR KG STATISTICS

Number of Node Types

```
MATCH (n)
RETURN count(DISTINCT labels(n)) AS num_node_types;
```

Number of Relationship Types

```
MATCH ()-[r]->()
RETURN count(DISTINCT type(r)) AS num_relationship_types;
```

Average Nodes per Profile

```
MATCH (c:Child)
WITH c, count { (c)--() } AS num_connected
RETURN avg(num_connected) + 1 AS avg_nodes_per_profile;
```

Average Edges per Child

```
MATCH (c:Child)
WITH c, count { (c)--() } AS edge_count
RETURN avg(edge_count) AS avg_edges_per_child;
```

Average Number of Flag Nodes per Profile

```
MATCH (c:Child)
WITH c, COUNT { (c)-[:HAS_FLAG]->(:Flag) } AS num_flags
RETURN avg(num_flags) AS avg_flag_nodes_per_child;
```

C AI DISCLOSURE

The system made use of LLM tools during development. The author takes full responsibility for all content, analysis and implementation. The model GPT 4.1 was used to develop the validation prompts, while the author adapted and refined the prompts according to the project goals and specifications. The model GPT o4-mini-high was used to assist in writing Cypher queries and Python scripts. No code was used verbatim, all code was critically reviewed, adapted and integrated into the system architecture by the author.