Scalable and Explainable Diet Recommendations via Answer Set Programming

Alina Vozna[1,2], Andrea Monaldini[1,2], Stefania Costantini[1] and Valentina Pitoni[1]

[1]Department of Information Engineering, Computer Science and Mathematics, University of L'Aquila, Italy,
[2]University of Pisa, Largo B. Pontecorvo, Pisa, 56127, Italy

Abstract

We present a logic-based approach to personalized dietary recommendations using Answer Set Programming (ASP). Our system represents user profiles, including dietary preferences, allergies, and nutritional needs, and selects compatible recipes that satisfy logical constraints while maximizing adherence to health goals. Through ASP, we ensure explainable reasoning and the ability to flexibly encode dietary rules. We illustrate the system's behavior through two representative case studies, involving complex user requirements and food restrictions. Furthermore, we conduct a scalability analysis by generating large sets of synthetic users and recipes, evaluating the system's performance under increasing data sizes. The results show that our ASP-based method remains interpretable and tractable for moderate-scale applications, with further optimization planned. This work contributes toward trustworthy and adaptable diet recommendation systems, paving the way for integration into personalized digital health solutions.

Keywords

Answer Set Programming (ASP), Cognitive Systems, Human-Robot Interaction (HRI), Socially Assistive Robotics (SARs), Blueprint Personas

1. Introduction

Personalized nutrition is emerging as a key factor in promoting long-term health and well-being. However, the task of generating individualized meal plans remains challenging due to the need to account for multiple constraints, such as dietary preferences, allergies, nutritional needs, and cultural factors. Moreover, systems that support such personalization should ideally be transparent, explainable, and adaptable to each user's profile.

In this work, we present a cognitively-inspired dietary advisor that combines user modeling through Blueprint Personas [1] with Answer Set Programming (ASP) [2, 3, 4, 5, 6] for reasoning and decision-making. Our system generates meal plans that are not only nutritionally appropriate but also compatible with the user's individual constraints.

While many multimedia recommendation systems have advanced significantly [7], food recommendation still lacks robust frameworks that integrate symbolic reasoning and cognitive personalization. Our contribution fills this gap by leveraging ASP for transparent reasoning and Blueprint Personas for structured cognitive modeling.

Unlike many black-box approaches that rely on machine learning, our solution uses ASP as a declarative formalism to represent knowledge, encode constraints, and compute valid plans in a fully explainable manner. Each user is modeled as a structured "Persona" encoding demographic, dietary, and health information. The system filters and assigns recipes based on this model and evaluates the compatibility and nutritional adequacy of the resulting plans.

HAIC 2025 - First International Workshop on Human-AI Collaborative Systems, October 25, 2025, Bologna, Italy.
Corresponding author.
These authors contributed equally.
alina.vozna@student.univaq.it (A. Vozna); andrea.monaldini@student.univaq.it (A. Monaldini); stefania.costantini@univaq.it (S. Costantini); valentina.pitoni@univaq.it (V. Pitoni)
0009-0009-0179-6948 (A. Vozna); 0009-0004-2518-2055 (A. Monaldini); 0000-0002-5686-6124 (S. Costantini); 0000-0002-4245-4073 (V. Pitoni)

In addition to its autonomous reasoning capabilities, the proposed ASP module is meant to be integrated into a socially assistive robot that interacts with users through a multimodal dialogue. This approach allows for an iterative cycle of refinement in which user feedback, contextual cues, and logical verification inform each other. The robot becomes not merely a prompter, but a collaborative partner that explains, adjusts, and co-constructs personalised meal plans with the user.

Our approach contributes to the broader field of Cognitive AI by integrating symbolic reasoning with cognitively inspired user modeling. We also outline possible extensions that involve natural language interaction and explainability mechanisms, paving the way toward more transparent and trustworthy human-AI collaboration in healthcare contexts.

The paper is organized as follows. In Section 2 we briefly discuss related work, and in Section 3 we provide some background on Answer Set Programming and Blueprint Personas. In Section 4 we illustrate the architecture of the proposed system, and in Section 5 the ASP engine. In Section 6 we discuss the issues of transparency and explainability. Section 7 presents two realistic case studies. In Section 8 we discuss limitations and perspectives of the present work, and, finally, in Section 9 we conclude.

2. Related Work

In recent years, the development of intelligent food recommendation systems has evolved significantly, moving beyond static preference matching toward more adaptive, personalized, and health-aware solutions. Various approaches have been proposed to tackle the complexity of dietary decision-making, integrating user preferences, nutritional constraints, contextual information, and long-term health goals. These systems leverage a broad range of AI techniques, from reinforcement learning and clustering to multi-agent architectures and fuzzy reasoning, to support users in achieving balanced, customized diets. The study [8] presents RecipeRL, an innovative food recommendation system that employs reinforcement learning to overcome the limitations of traditional systems. Unlike simple recommendation approaches, RecipeRL adopts a multi-step interaction model, dynamically adapting to changing user preferences and needs. The system introduces a representation of the user's state that fuses different information, such as interaction history and dynamic preferences, and exploits a simulated environment for training, demonstrating superior performance compared to conventional methods on real datasets.

By incorporating nutritional factors like calorie balance, the system in [9] seeks to address the drawbacks of conventional recommendation systems that just take user preferences into account. In order to recommend recipes that meet both individual taste and health requirements, the algorithm gathers information about the user's profile, long-term preferences, and sessional preferences. It also allows users to customise the relative weights of taste and health in the recommendations.

A recent approach to sustainable and health-aware food recommendation is HeaSE [10], which integrates nutrient-based retrieval, sustainability scoring, and large language models. The system first encodes recipes based on macro-nutrients, retrieves similar alternatives, and ranks them using a combined score derived from WHO nutritional guidelines and environmental impact (carbon and water footprint).

The authors in [11] emphasise how unhealthy diets contribute to the increase in non-communicable diseases and propose a model that, unlike previous work, handles nutritional aspects and preferences simultaneously. The proposed system includes a pre-filtering phase based on a multi-criteria decision analysis (AHPSort) to exclude inappropriate foods and a subsequent optimisation phase to generate daily food plans that are palatable to the user, not recently consumed, and nutritionally adequate. The aim is to provide more effective and personalised food recommendations to promote healthier eating habits.

The paper [12] presents Diet-Right, an intelligent cloud-based food recommendation system. The system uses an ant colony optimisation algorithm (ACO) to generate an optimal list of foods, with the aim of improving users' health through targeted food recommendations.

The work [13] presents a food recommendation system (FRS) specifically designed for people with diabetes. The main objective is to provide appropriate dietary suggestions that take into account the dietary limitations of these patients, going beyond simple recommendations based on the food pyramid. The system uses clustering analysis techniques, such as Self-Organising Maps (SOM) and K-means, to group

foods based on the similarity of eight nutrients that are significant for diabetics, thus allowing nutritionally relevant food substitutions to be suggested. The effectiveness of the system has been positively evaluated by nutritionists, highlighting its usefulness in the context of dietary management of diabetes.

An intelligent healthcare agent created to offer tailored meal suggestions related to the Taiwanese city of Tainan is presented in [14]. The agent suggests food routes to users based on their tastes and provides calorie information of suggested dishes displayed on Google Maps using a fuzzy inference mechanism, an ant colony optimisation technique, and a Tainan-specific food ontology.

The paper [15] introduces an ontology-based intelligent multi-agent system called IDFRMA (Intelligent Diet Food Recommendation Multi-Agent), with the aim of providing personalised and healthy dietary recommendations. The system uses different types of agents, each specialising in a specific task such as analysing the user profile, nutritional values, and applying fuzzy logic to assess diet status. The use of ontologies allows knowledge about food and nutritional needs to be represented in a structured way, enabling the agents to reason semantically and offer a detailed and comprehensible analysis of the user's eating habits.

Our approach differs from the above works, as it relies on symbolic reasoning through Answer Set Programming, offering full explainability, constraint-based safety, and seamless integration within a socially-assistive robotic platform. In contrast to heuristic or sub-symbolic methods such as reinforcement learning, ant colony optimization, fuzzy systems, or multi-agent architectures, ASP allows knowledge and preferences to be explicitly represented and verified. This supports transparent decision-making, facilitates user trust, and simplifies adaptation to new dietary rules or user profiles without retraining. Such features are particularly valuable in health-related contexts, where traceability and personalization are essential

3. Background

3.1. Answer Set Programming and Answer Set Semantics

In Answer Set Programming (ASP), one can see a program as a set of statements that specify a problem, where each answer set represents a solution compatible with this specification. Sometimes, in fact, an ASP program has no answer sets, i.e., no solution can be found, and the program/specification is said to be "inconsistent". Several well-developed freely available *answer set solvers* exist [16], that compute all existing solutions. Syntactically, an ASP program is a collection of *rules* of the form

```
A_1 | ... | A_g :- L_{1} , ... , L_{n}.    where  g >= 1, 0 <= i <= n
```

Each A in the rule is an atom, "|" indicates disjunction, and the L's are literals (i.e., atoms or negated atoms of the form not A). The left-hand side and the right-hand side of the rule are called *head* and *body*. A rule with an empty body is called a *fact*. Disjunction can occur in rule heads only, and so also in facts.

A rule with empty head (or, equivalently, with head "false"), of the form

```
:- L_1,...,L_n.
```

is a *constraint*, stating that the literals occurring therein are not allowed to be simultaneously true in any answer set; the impossibility to fulfil such a requirement is one of the reasons that can make a program inconsistent.

Programming methodology in ASP is of the generate-and-test kind: a part of the program, composed of rules, generates possible solutions; another part of the program, composed of constraints, filters the acceptable ones, ruling out the others.

The answer set (or "stable model'') semantics, on which answer set solvers are based, can be defined in several ways [17, 18]. However, the answer sets of a program, if any exist, are the supported minimal classical models of the program interpreted as a first-order theory in an obvious way.

3.2. Blueprint Personas as Cognitive User Models

The European Blueprint on Digital Transformation of Health and Care for the Aging Society, funded as part of the WE4AHA project under the European Commission's digital single market strategy, reflects a shared policy vision on how innovation can transform health and care provision in our aging society. The first time it was presented was in December 2016 at the EIP on AHA Conference [19]. It was described as a shared policy vision by many stakeholders, including policymakers, civil society, professional organizations, and industry, in the first update in 2017. This marks a new strategy for securing funding and commitments for the digital transformation of health and care [20].

Blueprint Personas are used to encode cognitively plausible user profiles. Each persona includes demographic attributes (age, gender), lifestyle characteristics, dietary goals, and health-related constraints. Unlike static profiles, Blueprint Personas offer a structured format that reflects user diversity and supports symbolic reasoning.

In our framework, personas are used as the basis for instantiating user models in the ASP program. For example, a "persona" with a vegetarian diet and lactose intolerance will only be matched with recipes that satisfy both constraints. This allows the system to simulate reasoning over different user types and adapt recommendations accordingly.

We argue that Blueprint Personas support a form of computational Theory of Mind [21, 22], allowing the system to represent and reason about user needs in a human-like way. This aligns with the goals of Cognitive AI, where systems are expected to act not just rationally, but also socially and contextually.

4. System Overview

The architecture of our dietary advisor consists of three main components: (i) user modeling with Blueprint Personas, (ii) logic-based reasoning with Answer Set Programming, and (iii) menu generation with nutritional evaluation. Users and recipes are represented in structured databases. Each recipe includes information about ingredients, nutritional values, dietary labels (e.g., vegetarian, gluten-free), and potential allergens.

The recommendation process starts by filtering out recipes that are incompatible with the user's dietary restrictions. ASP is then used to assign one compatible recipe for each meal course (e.g., starter, main dish, side, dessert), ensuring that all constraints are satisfied. Additionally, nutritional totals (e.g., calories, proteins, carbohydrates, fats) are calculated and used to evaluate the adequacy of the suggested plan.

The system supports multiple users with heterogeneous needs. Each user is processed independently, and the solver explores valid recipe combinations to propose balanced and individualized menus. Thanks to the declarative nature of ASP, we can easily express global constraints (e.g., total calories within a range) and logical dependencies (e.g., meals cannot contain allergens).

To support both reasoning and large-scale testing, users and recipes are modeled as structured ASP facts. Each user includes demographic data, nutritional needs, and optional constraints such as allergies, vegetarianism, or dislikes. Recipes contain metadata such as course type, list of ingredients, macronutrient values, and dietary labels. Below is a simplified example of the synthetic data schema used in our experiments:

```
% User profile
user("user0", 35, "female").
daily_needs("user0", 2200, 90, 300, 70).
vegetarian("user0", true).
allergy("user0", "gluten").
dislikes("user0", "carote (carrot)").

% Recipe definition
recipe("r1", "recipe_1", "main_course").
vegetarian("r1", false).
contains("r1", "pasta").
contains("r1", "pomodoro (tomato)").
```

```
contains_allergen("r1", "gluten").
macronutrients("r1", 500, 20, 60, 15).
```

Listing 1: Example of ASP facts representing users and recipes.

This structure allows the ASP reasoning engine to filter and assign recipes while supporting constraint validation and aggregation. Synthetic data for scalability experiments were generated using a randomized generator that populates these predicates with realistic values and distributions (see Section **??**).

4.1. Integration within a Cognitive Architecture for Social Robots

In the broader context of trustworthy and explainable human-robot interaction, our ASP-based dietary advisor is designed to operate as the Verification Module within a cognitively-inspired architecture for Socially Assistive Robots (SARs), as introduced in [23]. This architecture integrates perception, memory, decision-making, and multimodal interaction to support personalized and transparent dietary coaching, especially for patients with health-related dietary needs.

In particular, our ASP logic is invoked after the proposal of a set of preliminary recommendations to the user, elaborated by the LLM-based Decision Module. The ASP program refines this output by applying domain-specific constraints, encoded in a declarative and explainable way.

To ensure transparency, the ASP output is mapped to an explainable ”inner speech” prompt, which can be verbalized by the robot using an LLM-based generation mechanism. This enables users to follow the robot's reasoning process and develop a sense of trust and collaboration.

By embedding our reasoning system within this cognitively-inspired architecture, we ensure that dietary recommendations are not only personalized and nutritionally sound but also cognitively explainable, socially aware, and robotically actionable.

5. Answer Set Programming for Dietary Reasoning

ASP serves as the core reasoning engine for recipe filtering, assignment, and validation. We encode constraints to eliminate incompatible recipes and apply choice rules to assign exactly one recipe per course. The declarative nature of ASP allows the solver to explore all valid combinations, supporting flexibility and completeness.

Each recipe is annotated with macronutrient information, and ASP aggregates these values to compute daily totals. Constraints are applied to ensure that the resulting plans are within acceptable nutritional ranges (e.g., 1500–2200 kcal per day). Additional logic ensures that the menu is complete, meaning that each user receives a starter, main course, side dish, and dessert.

By combining symbolic constraints with structured user models, our system ensures logical consistency and explainability in the recommendation process.

5.1. Dietary Constraints and Compatibility Logic

The ASP program incorporates domain-specific knowledge through a set of constraints that define dietary compatibility. These rules ensure that the assigned recipes are suitable for each user based on their dietary restrictions and lifestyle choices. The key constraints include:

- Allergy Filtering: Users with lactose or gluten intolerance are automatically excluded from recipes containing those ingredients.
- Vegetarian Compliance: Vegetarian users are only matched with vegetarian recipes, while non-vegetarians can consume any dish.
- Course Completion: Each user must receive one recipe per course (main, second, side dish, dessert) to form a complete daily meal.
- Recipe Assignment: The use of choice rules allows ASP to select one compatible recipe per course for each user, considering all valid combinations.

- Nutritional Aggregation: Macronutrient values are summed across selected dishes to compute totals, which are used for further validation or display.

To ensure robustness, the ASP program only requires a course assignment when at least one compatible recipe exists. If no valid recipe is available for a given user and course (due to strict constraints or missing options), no assignment is forced. This fallback mechanism allows the system to return partial but consistent meal plans rather than invalid ones, ensuring graceful degradation in constrained scenarios.

```
:- user(Name, _, _), dish_type(Course), not 1 { assign(Name, _, Course) } 1, 1 { recipe(R, _,
    Course) : compatible(Name, R) }.
```

Listing 2: Meal Completion Constraints

Before recipe assignment, a set of compatibility predicates (*lactose_compatible/2*, *gluten_compatible/2*, *vegetarian_compatible/2*) ensures that recipes do not conflict with the user's dietary profile. These predicates are used to define the general recipe compatibility rule:

```
intolerance(Name, Allergen) :- allergy(Name, Allergen).

has_conflicting_allergen(Name, RecipeID) :-
    intolerance(Name, Allergen),
    contains_allergen(RecipeID, Allergen).

has_disliked_ingredient(Name, RecipeID) :-
    dislikes(Name, Ingredient),
    contains(RecipeID, Ingredient).

compatible(Name, RecipeID) :-
    user(Name, _, _),
    recipe(RecipeID, _, _),
    vegetarian_compatible(Name, RecipeID),
    not has_conflicting_allergen(Name, RecipeID),
    not has_disliked_ingredient(Name, RecipeID).
```

Listing 3: Compatibility Constraint

The core choice rule for assigning recipes is defined as follows:

```
{ assign(Name, RecipeID, Course) } :-
    user(Name, _, _),
    recipe(RecipeID, _, CourseType),
    compatible(Name, RecipeID).
```

Listing 4: Choice Rule

This is a choice rule that enables the ASP solver to non-deterministically select a compatible recipe for each user, for every course type. The rule is guarded by the *compatible/2* predicate, which ensures that only recipes satisfying all dietary constraints—such as allergies and dietary preferences—are considered valid candidates. In conjunction with the constraints that enforce the assignment of exactly one recipe per course, this rule guarantees both the correctness and completeness of the resulting meal plan, allowing the solver to explore multiple valid combinations while respecting the user's individual requirements.

5.2. Nutritional Inference and Aggregation

In addition to filtering and assigning recipes, the system computes the aggregate nutritional intake for each user. This allows for a posteriori evaluation of the dietary adequacy of the generated meal plan. For each assigned recipe, the system extracts macronutrient information—calories, proteins, carbohydrates, and fats—and sums these values across the selected courses. This inferential step enables a transparent and

interpretable summary of the dietary composition, which can later be used for verification, visualization, or further optimization.

Currently, the nutritional values in our system refer to pre-defined standard portions of each recipe, as specified in the underlying dataset. However, these values are not yet dynamically adjusted based on specific user needs or serving sizes. Future extensions will include the ability to compute and recommend appropriate portion quantities, allowing for more precise alignment with individual nutritional goals (e.g., recommending 250g of bean soup to meet a 21g protein target).

The nutritional totals are computed through the following aggregate rules.

```
person(Name) :- user(Name, _, _).

total_calories(Name, Total) :- person(Name), Total = #sum { C : assign(Name, R, _),
    macronutrients(R, C, _, _, _) }.
total_proteins(Name, Total) :- person(Name), Total = #sum { P : assign(Name, R, _),
    macronutrients(R, _, P, _, _) }.
total_carbs(Name, Total) :- person(Name), Total = #sum { K : assign(Name, R, _),
    macronutrients(R, _, _, K, _) }.
total_fats(Name, Total) :- person(Name), Total = #sum { F : assign(Name, R, _),
    macronutrients(R, _, _, _, F) }.
```

Listing 5: Macronutrient Aggregation Rules

These results reflect how Blueprint Personas can be used not only to filter recipes by compatibility but also to verify that the generated meal plans align with cognitive and health-related constraints. In a future extension, these summaries may be vocalized or visualized via a robotic interface, increasing user trust and fostering more informed choices, particularly for vulnerable or elderly individuals.

5.3. User Interface

To operationalize the logic-based verification process described above, we developed a Python interface using Clingo 5.7.1 [1]. This component acts as a middleware between the ASP-based verification layer and the robot or human-facing modules. Its purpose is to execute the ASP program, interpret the results, and classify the outcome into one of the three categories described:

- No valid plan found: the robot triggers an adaptive persuasion strategy.
- Single valid plan: the robot provides a verbal explanation based on logical traceability.
- Multiple valid plans: the user is engaged in a dialogue to refine the choice.

The Python script acts as a bridge between the symbolic reasoning component and the robotic interface. It loads the ASP program and performs grounding using Clingo, initializing the reasoning process with a randomized seed and allowing up to ten valid answer sets to be generated through stochastic choice rules. Once the solutions are computed, the script extracts key predicates such as *user_info/5*, *suggest/3*, and the various *total_** values. These data are then structured into a human-readable summary, which can be either displayed to the user or verbalized by the robot to support explainability and transparency in the recommendation process.

6. Cognitive Interpretation and Transparency

In health-related recommendation scenarios, especially those involving dietary guidance, the ability to interpret and explain a system's reasoning is essential for user trust, compliance, and engagement. Our ASP-based framework provides a clear cognitive advantage by enabling explicit, logic-based explanations of recipe assignments and nutritional evaluations. Each step of the reasoning process, from compatibility filtering to final menu assignment, is encoded in interpretable logical rules. This allows the system to not

[1] https://github.com/potassco/clingo/releases/tag/v5.7.1

only generate optimized meal plans but also to justify each recommendation in human-understandable terms.

By design, ASP supports symbolic transparency: the constraints, preference rules, and nutritional aggregations are expressed declaratively, and their outcomes can be directly inspected and verbalized. For example, a user with lactose intolerance receives a meal plan that omits dairy-containing recipes, not through a black-box classification but via explicitly enforced constraints. Similarly, macronutrient totals can be explained as the sum of the assigned dishes, each traceable to a known data source.

These properties support the integration of a cognitive layer into the system architecture, where internal reasoning traces can be externalized through verbal or visual feedback. This concept aligns with prior work on inner speech in social robots, where self-generated explanations improve transparency, promote trust, and simulate human-like introspection. Although our current system does not yet generate natural language self-dialogue, the logical derivations already represent a form of "cognitive trace" that can be exposed to users through multimodal interfaces.

Example explanation dialogue (future integration):

> "I assigned you *zuppa di fagioli (bean soup)* as your main course because it is vegetarian and lactose-free, which matches your profile. The total protein content of your plan is 21 grams, which aligns with your daily needs."

This ability is especially important when used in combination with Blueprint Personas, which model not only dietary restrictions but also psychological, behavioral, and motivational factors. By grounding explanations in the user's persona (e.g., "Diana is a lactose-intolerant vegetarian with moderate protein requirements"), the system can deliver personalized justifications that are context-aware and cognitively relatable. This strengthens user engagement, supports ethical AI practices, and opens the path toward explainable human-robot dietary coaching in real-world settings.

Recent advances in Large Language Models (LLMs) open promising avenues to extend the cognitive capabilities of our system. By coupling symbolic reasoning with LLM-generated natural language, the system could dynamically transform its logical explanations into human-like dialogues. For example, the symbolic justification behind the exclusion of a lactose-containing recipe can be translated into an interactive explanation such as:

> "Since you are lactose intolerant, I did not include panna cotta (Italian cooked cream dessert) in your dessert options. Instead, I selected fresh fruit, which is safer and aligns with your dietary needs."

This form of verbalized inner reasoning has already been shown to improve explainability and engagement in social robots [24]. Integrating LLMs as an outer speech module could allow the system to disclose its decision-making process using context-aware responses that align with the user's persona and cognitive profile. Such multimodal, explainable interactions would make dietary planning not only transparent and personalized but also socially intelligible and cognitively grounded.

The system also explains when a recipe is excluded due to voluntary ingredient preferences:

> "I did not include risotto ai funghi (mushroom risotto) because you mentioned you don't like mushrooms."

This enhances transparency and reflects a more human-like understanding of dietary habits. While we do not perform a direct empirical comparison with fuzzy logic or machine learning-based diet recommendation systems, it is important to highlight some qualitative distinctions. Black-box approaches, including neural models and optimization heuristics, may excel in large-scale accuracy metrics, but they often lack explainability and fine-grained constraint control. In contrast, our ASP-based system offers full transparency, logical traceability, and verifiable safety, all of which are critical in health-related scenarios where user trust and ethical oversight are essential. Furthermore, symbolic reasoning enables easy adaptation to new user types and domain knowledge, without retraining or loss of interpretability.

7. Case Study and Examples

To illustrate the effectiveness of our ASP-based dietary recommendation framework, we present two realistic case studies derived from instantiated Blueprint Personas. Each user is characterized by demographic attributes, dietary preferences, and nutritional goals, and the system is tasked with generating a full meal plan satisfying their constraints. The solver must ensure course completeness, compatibility with allergies and dietary style, and macronutrient aggregation

7.1. Persona 1: Diana – Young Vegetarian with Lactose Intolerance

Diana is a 28-year-old female who follows a vegetarian diet and is allergic to lactose. The system uses this profile to assign one recipe per course that complies with her dietary constraints.

```
% USERS
user("Diana", 28, "female").
daily_needs("Diana", 2200, 75, 270, 70).
allergy("Diana", "lactose").
vegetarian("Diana", true).
```

Listing 6: ASP Facts – Blueprint Persona: Diana

The ASP solver filters out all non-vegetarian recipes as well as those that contain lactose. From the remaining pool of recipes, one compatible option per course type is selected. Once the plan is generated, the system calculates the total nutritional intake and presents it alongside the meal suggestions.

```
=== USER: "Diana" ===•
 Age: 28, Gender: "female"•
 Diet: "vegetarian", Allergies: "lactose"

MEAL PLAN:
- Main Course: zuppa di fagioli (bean soup)
- Second Course: frittata di zucchine (zucchini omelette)
- Side Dish: crostini vegetali (vegetable crostini)
- Dessert: frutta fresca (fresh fruit)

NUTRITIONAL TOTALS:•
 Calories: 1000 kcal•
 Proteins: 21 g•
 Carbohydrates: 125 g•
 Fats: 30 g
```

Listing 7: System Output – Diana's Recommended Meal Plans

This result reflects how the system satisfies the compatibility constraints while producing a nutritionally balanced and coherent plan. Although Diana's full daily needs are not met, the plan represents a safe, consistent base that could be expanded with additional portions or snacks.

7.2. Persona 2: Eve – Non-Vegetarian with Gluten Intolerance

Eve is a 22-year-old female with no dietary restrictions but a strong intolerance to gluten. Her nutritional goals are slightly lower than Diana's, with 1900 kcal, 65 grams of protein, 240 grams of carbohydrates, and 55 grams of fat. The system must filter out recipes containing gluten while preserving access to meat- or fish-based dishes.

```

=== USER: "Eve" ===•
 Age: 22, Gender: "female"•
 Diet: "non-vegetarian", Allergies: "gluten"

```

```
MEAL PLAN:
- Main Course: crema di zuppa di pomodoro (creamy tomato soup)
- Second Course: pollo al curry (chicken curry)
- Side Dish: zucchine grigliate (grilled zucchini)
- Dessert: sorbetto al limone  (lemon sorbet)

NUTRITIONAL TOTALS:•
 Calories: 920 kcal•
 Proteins: 46 g•
 Carbohydrates: 75 g•
 Fats: 20 g
```

Listing 8: System Output – Eve's Recommended Meal Plans

Eve's result shows a high-protein selection that avoids all gluten-containing recipes. While the total energy intake is below the target threshold, the solver prioritizes safety and compatibility. The plan can be complemented with additional gluten-free snacks or adjusted through future iterations.

7.3. Scalability Experiments and System Performance

To evaluate the scalability and efficiency of our ASP-based recommendation system, we conducted a series of controlled experiments by varying the number of generated users and recipes. All user and recipe data were synthetically generated using a Python script, which simulates realistic values and constraints.

Each user is generated with the following attributes:

- **Age and gender** randomly sampled within realistic adult ranges.
- **Daily nutritional needs** (calories, proteins, carbs, fats) calculated using standard formulas.
- **Optional features** such as *vegetarian status*, *food allergies*, and *dislikes* are added randomly with fixed probabilities (e.g., 1 out of 3 users is vegetarian or allergy-free).

Recipes are also generated programmatically, each assigned to a course type (*main_course*, *dessert*, *second_course*, etc.), and populated with random combinations of ingredients. Macronutrients are assigned proportionally to ingredients, and each recipe is marked as vegetarian or not. Known allergens are derived from ingredients, ensuring the logic engine has complete symbolic knowledge of each element.

The experiment setup varies the number of users and recipes across a range of values, and for each configuration, the time required to compute a valid meal assignment is measured. The Python engine builds the ASP facts and invokes Clingo 5.7.1, measuring the time from grounding to solution retrieval.

Table 1 shows the execution times (in seconds) for each configuration of users and recipes.

Figure 1 visualizes the increase in execution time with respect to the number of users and recipes. As expected, the growth is non-linear due to the combinatorial nature of the search space. The number of recipes plays a particularly significant role: doubling the recipe count can result in a 2-5x increase in computation time even with the same number of users.

These results confirm that while the problem is computationally demanding due to hard constraints and cross-dependencies, the system remains tractable even with realistic user volumes. This validates the applicability of logic-based approaches to personalized recommendation in domains where correctness and explainability are critical.

8. Discussion and Future Work

The results of our evaluation demonstrate the expressiveness and practical viability of leveraging Answer Set Programming (ASP) for personalized diet recommendation tasks. Through both qualitative case studies and controlled scalability tests, we observed that our approach effectively handles diverse dietary constraints, user preferences, and clinical needs. In particular, the modular knowledge base and reasoning capabilities of ASP enabled explainable, constraint-aware, and personalized meal assignment, aligning with the core goals of trust and transparency in health-related AI systems.

Table 1
Execution time (in seconds) as a function of user and recipe count

Users	10 Recipes	20 Recipes	40 Recipes
10	0.1544	36.361	212.997
20	0.1622	78.425	340.554
50	0.3785	235.572	1035.965
100	18.280	517.641	2649.487
200	26.910	610.988	7574.681

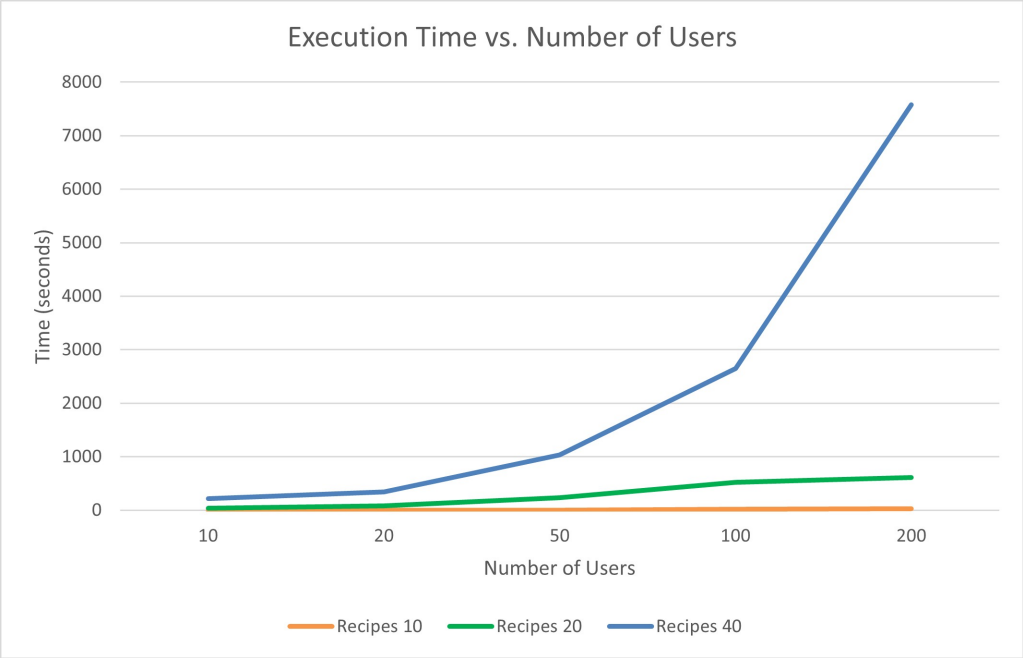

Figure 1: Execution time vs. number of users

However, several limitations and opportunities for future improvement emerged during our development and experimentation. First, realistic user modeling will be addressed by replacing the current synthetic data with real-world dietary datasets, such as hospital records, nutrition tracking apps, and wearable-derived data (e.g., calorie intake and physical activity), to enhance ecological validity. Regarding interaction and adaptation, the current system is static and logic-driven, but its integration with interactive components, such as conversational agents or affect-aware robots, would enable dynamic adaptation to user feedback, mood, or contextual changes. This vision aligns with recent efforts to embed ASP within hybrid cognitive architectures that combine symbolic and sub-symbolic reasoning layers. Although ASP provides symbolic justifications by design, further work is needed to translate these reasoning steps into human-understandable explanations through natural language feedback or visual justifications. This aspect is particularly important in sensitive healthcare scenarios involving vulnerable users. Currently, our system assumes that each user receives a full meal with all four courses (starter, main, side, dessert) at every dining occasion. While this provides a standardized baseline for evaluation, we recognize that future versions should allow users to personalize their meal structure, for example, by skipping dessert at dinner but including it at lunch. This would enable more accurate modeling of real-world preferences and facilitate better nutritional distribution across the day.

While our current implementation demonstrates that the ASP-based system is effective up to moderate scales, the observed non-linear increase in computation time suggests challenges in real-world deployments such as hospitals, where hundreds of patients and thousands of recipes may be involved. This is a known limitation of logic-based global optimization under hard constraints. Future work will investigate modularization strategies, such as partitioning the user base into cohorts, using incremental or distributed

solving, and integrating approximate or heuristic-based pre-filtering to reduce the grounding size before solving. These extensions are essential for enabling scalable deployment in large institutions.

Finally, we aim to explore the generalization of our logic-based recommendation strategy to other domains that require complex multi-constraint reasoning, such as medication planning, care task scheduling, or educational guidance.

Overall, this work provides a solid foundation for scalable, explainable, and adaptable reasoning in personalized health recommendation systems. We envision the integration of ASP with perceptual modules and large language models as a promising direction to further enhance the responsiveness and empathy of AI-driven assistive technologies.

9. Conclusions

In this work, we proposed an ASP-based framework for personalized dietary recommendation, integrating formal logic, user profiling, and constraint reasoning. Our system models individual preferences, allergies, nutritional needs, and lifestyle constraints, generating tailored meal suggestions that are both nutritionally adequate and explainable.

We validated our approach through two realistic case studies and a scalability experiment involving synthetic users and recipes. Results show that ASP enables high-level expressiveness and flexibility in modeling heterogeneous users, while maintaining control over logical correctness and explainability. The final performance evaluation demonstrated that, although reasoning time grows with problem size, the system remains feasible for medium-scale applications and offers a promising foundation for assistive technologies.

By leveraging the declarative nature of ASP, our system is transparent, auditable, and adaptable, key features for AI applications in healthcare domains where user trust, safety, and personalization are critical.

This work sets the stage for future extensions, including integration with perception modules, real-world datasets, and user feedback loops. Ultimately, it contributes to the growing field of symbolic AI for trustworthy and human-centered recommendation systems.

Declarations

During the preparation of this work, the authors used Grammarly and ChatGPT (OpenAI) to assist with grammar and spelling checks, as well as paraphrasing and rewording of certain sentences. After using these tools, the authors carefully reviewed and edited the content to ensure accuracy, coherence, and compliance with academic standards. The authors take full responsibility for the content of this publication.

Acknowledgments

Research partially supported by the PNRR Project CUP E13C24000430006 ”Enhanced Network of intelligent Agents for Building Livable Environments - ENABLE”, and by PRIN 2022 CUP E53D23007850001 Project TrustPACTX - Design of the Hybrid Society Humans-Autonomous Systems: Architecture, Trustworthiness, Trust, EthiCs, and EXplainability (the case of Patient Care), and by PRIN PNNR CUP E53D23016270001 ADVISOR - ADaptiVe legIble robotS for trustwORthy health coaching. In addition, we acknowledge the support from Progetto PRIN 2022 PNRR “ADVISOR – ADaptiVe legIble robotS for trustwORthy health coaching”

References

[1] A. Monaldini, A. Vozna, S. Costantini, Blueprint personas in digital health transformation (2024).
[2] G. Brewka, T. Eiter, M. T. (eds.), Answer set programming: Special issue, AI Magazine 37 (2016).

[3] C. Baral, Knowledge representation, reasoning and declarative problem solving, Cambridge university press, 2003.
[4] S. Costantini, A. Formisano, Answer set programming with resources, Journal of Logic and Computation 20 (2010) 533–571.
[5] M. Gelfond, Answer sets, in: Handbook of Knowledge Representation. Chapter 7, Elsevier, 2007.
[6] V. Lifschitz, Answer set programming, volume 3, Springer Cham, 2019.
[7] W. Min, S. Jiang, R. Jain, Food recommendation: Framework, existing solutions, and challenges, IEEE Transactions on Multimedia 22 (2019) 2659–2671.
[8] L. Liu, Y. Guan, Z. Wang, R. Shen, G. Zheng, X. Fu, X. Yu, J. Jiang, An interactive food recommendation system using reinforcement learning, Expert Systems with Applications 254 (2024) 124313.
[9] M. Ge, F. Ricci, D. Massimo, Health-aware food recommender system, in: Proceedings of the 9th ACM Conference on Recommender Systems, 2015, pp. 333–334.
[10] A. Petruzzelli, C. Musto, M. C. Di Carlo, G. Tempesta, G. Semeraro, Recommending healthy and sustainable meals exploiting food retrieval and large language models, in: Proceedings of the 18th ACM Conference on Recommender Systems, 2024, pp. 1057–1061.
[11] R. Y. Toledo, A. A. Alzahrani, L. Martinez, A food recommender system considering nutritional information and user preferences, IEEE Access 7 (2019) 96695–96711.
[12] F. Rehman, O. Khalid, K. Bilal, S. A. Madani, et al., Diet-right: A smart food recommendation system, KSII Transactions on Internet and Information Systems (TIIS) 11 (2017) 2910–2925.
[13] M. Phanich, P. Pholkul, S. Phimoltares, Food recommendation system using clustering analysis for diabetic patients, in: 2010 international conference on information science and applications, IEEE, 2010, pp. 1–8.
[14] C.-S. Lee, M.-H. Wang, W.-C. Sun, Y.-C. Chang, Intelligent healthcare agent for food recommendation at Tainan city, in: 2008 IEEE International Conference on Systems, Man and Cybernetics, IEEE, 2008, pp. 1465–1470.
[15] K. Prakash, R. Sivakumar, Ontology-based intelligent multi-agent for diet food recommendation, International Journal of Scientific & Engineering Research 4 (2013) 1–8.
[16] Web references of ASP solvers, 2025. Clingo: potassco.org; Cmodels: www.cs.utexas.edu/users/tag/cmodels; DLV: www.dlvsystem.it; WASP: alviano.github.io/wasp.
[17] V. Lifschitz, Thirteen definitions of a stable model, in: A. Blass, N. Dershowitz, W. Reisig (Eds.), Fields of Logic and Computation, Essays Dedicated to Yuri Gurevich on the Occasion of His 70th Birthday, volume 6300 of *LNCS*, Springer, 2010, pp. 488–503. doi:10.1007/978-3-642-15025-8_24.
[18] S. Costantini, A. Formisano, Negation as a resource: a novel view on answer set semantics, Fundamenta Informaticae 140 (2015) 279–305.
[19] C. E., European innovation partnership on active and healthy ageing, https://ec.europa.eu/eip/ageing/home_en.html, 2024. Accessed: 2024-09-23.
[20] E. Commission, Blueprint for digital transformation of health and care in an ageing society, https://digital-strategy.ec.europa.eu/en/library/blueprint-digital-transformation-health-and-care-ageing-society, 2024. Accessed: 2024-09-23.
[21] D. Premack, G. Woodruff, Does the chimpanzee have a theory of mind?, Behavioral and brain sciences 1 (1978) 515–526.
[22] L. Camaioni, P. Di Blasio, Psicologia dello sviluppo, Società editrice il Mulino, Spa, 2020.
[23] L. D'Arco, L. Raggioli, G. Randazzo, G. De Gasperis, A. Chella, S. Costantini, S. Rossi, Towards trustworthy and explainable socially assistive robots: A cognitive architecture for dietary guidance, in: 2025 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR), IEEE, 2025, pp. 1–6.
[24] S. Stange, T. Hassan, F. Schröder, J. Konkol, S. Kopp, Self-explaining social robots: An explainable behavior generation architecture for human-robot interaction, Frontiers in Artificial Intelligence 5 (2022) 866920.