Personalized Supplement Recommender Using Agents and Knowledge Graphs

Jovanna Fernando
jfernando@ucsd.edu

Kevin Chen
kec050@ucsd.edu

Parna Praveen
ppraveen@ucsd.edu

Sia Patodia
spatodia@ucsd.edu

Abed El-Husseini
aelhusseini@deloitte.com

Balaji Veeramani
baveeramani@deloitte.com

Abstract

The dietary supplement industry poses significant safety risks for consumers. Supplements frequently interact with prescription medications in potentially dangerous ways, and many medications deplete essential nutrients, yet these risks often go unrecognized. Existing tools rely on static lookups or general-purpose language models that lack the structured biomedical knowledge needed for personalized, evidence-grounded guidance. This project addresses these gaps with an AI-powered supplement safety advisor that analyzes individual health profiles - including medications, dietary restrictions, medical conditions, and symptoms - to identify contraindicated supplements, detect nutrient deficiencies, and surface safe, personalized candidates. Our approach combines a Neo4j knowledge graph encoding biomedical relationships from trusted sources (DrugBank, Mayo Clinic, NIH, MedlinePlus) with a multi-agent LangGraph pipeline powered by Claude. The knowledge graph represents medications, supplements, nutrients, and conditions with relationships including drug-supplement interactions, medication-induced nutrient depletion, and contraindications, enabling multi-hop reasoning to detect complex risks that simple lookup tools miss. All outputs are grounded in explicit graph evidence and delivered in plain language accessible to non-expert users. By making supplement recommendations logical and evidence based, this project can help users make safer, more informed decisions about their health.

Code: GitHub Repository

Website: Project Website

Demo: Live Demo

1 Introduction

1.1 Problem Statement and Background

The dietary supplement industry has grown into a roughly $150 billion market in the United States, with millions of Americans taking vitamins, minerals, and herbal products daily to improve their health. Unlike prescription medications, dietary supplements do not require FDA approval before entering the market and do not need to demonstrate safety or efficacy through clinical trials. This regulatory gap has created a landscape where products can make vague health claims such as "supports immunity" or "boosts energy" without scientific backing, leaving consumers in a market driven more by marketing than evidence.

This can result in serious health consequences as supplements interact with common medications in ways most consumers do not anticipate. For example, Fish Oil can amplify the blood-thinning effects of Warfarin, increasing bleeding risk - an interaction our system correctly identifies across multiple reasoning pathways. High-dose Vitamin K similarly undermines Warfarin's efficacy by counteracting its anticoagulant mechanism. The people most vulnerable to these risks are often those trying hardest to improve their health: individuals managing chronic conditions, older adults on multiple medications, and health-focused consumers attempting to make informed decisions. Without personalized guidance that considers their complete health picture, these consumers face the difficult task of manually gathering information on drug interactions, nutrient depletions, dietary restrictions, and individual health conditions.

This project demonstrates how combining Large Language Models with agentic reasoning frameworks and structured biomedical knowledge graphs can address these gaps - delivering personalized, evidence-grounded supplement safety guidance that is traceable, accurate, and accessible to non-expert users.

1.2 Literature Review and Prior Work

The biomedical AI field has changed dramatically with the introduction of systems that can plan, reason, and execute tasks autonomously. Unlike earlier models that produced results without explanation, modern agentic frameworks combine language model reasoning with external tool integration. The ReAct paradigm (Yao et al. 2022) started this approach, allowing AI to alternate between thinking and action to break complex problems into manageable steps. This has led to domain-specific applications such as CRISPR-GPT (Qu et al. 2024), which handles gene-editing workflows, and Biomni (Huang et al. 2024), which traverses knowledge graphs across diverse research tasks. These systems demonstrate that when agents are deployed in well-defined biomedical domains with tailored tooling, they can automate complex technical workflows effectively.

Existing tools for supplement safety each have fundamental limitations. Platforms like Examine.com provide evidence-based literature on supplement efficacy, safety, and drug interactions, but their analysis is supplement-centric rather than patient-centric — users

must still manually integrate their full health profile, medications, and dietary restrictions to arrive at a personalized safety assessment. Interaction-checking tools such as the Natural Medicines Database and Drugs.com operate as static lookup systems: users input a medication list and receive binary flags for documented interactions, but these tools cannot incorporate dietary restrictions, reason across indirect pathways, or maintain context across a complete health profile.

The gap in existing work is the combination of comprehensive biomedical knowledge with personalized, multi-hop reasoning across relationship types. Current tools either provide generic information or perform limited automated checks that miss indirect risks. This project addresses that gap by integrating biomedical graph querying and agentic reasoning into a unified workflow - enabling interpretable, safety-critical guidance personalized to each user's complete health profile.

1.3 Description of Relevant Data

The knowledge graph integrates three primary data sources to create a comprehensive biomedical knowledge base: DrugBank for pharmaceutical information, Mayo Clinic for supplement data, and curated nutritional datasets from MedlinePlus and NIH for dietary restriction analysis. The graph contains 329,849 nodes and 3,447,235 relationships spanning medications, supplements, nutrients, dietary patterns, and their complex interactions.

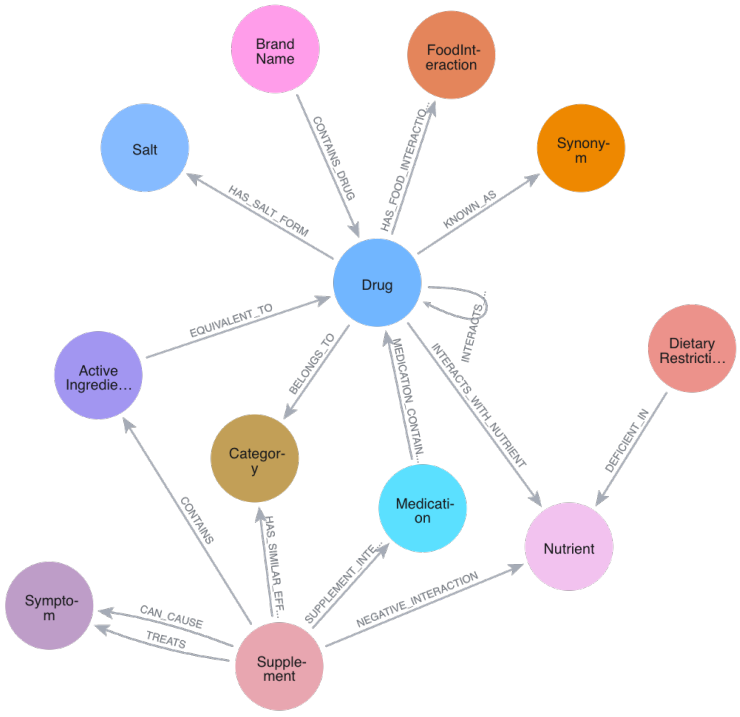


Figure 1: Knowledge graph schema illustrating core node types and cross-domain relationships across medication, supplement, and nutritional domains.

1.3.1 Knowledge Graph Nodes

The knowledge graph represents biomedical entities across pharmaceutical, supplement, and nutritional domains. Node types are grouped by domain to reflect their functional role in downstream reasoning.

Drug Domain Nodes These nodes represent prescription medications and related pharmaceutical concepts used for interaction and safety analysis.

- **Drug** nodes represent prescription medications and include attributes such as drug name (e.g., *Warfarin*, *Metformin*), unique drug identifiers, indication text, mechanism of action, toxicity profile, and drug type classification.
- **Category** nodes capture therapeutic and chemical classifications (e.g., Anticoagulants, Statins, Antidiabetic Agents) that group drugs by mechanism of action or clinical use.
- **BrandName** nodes represent commercial formulations with brand names, dosage forms, strengths, and routes of administration.
- **Salt** nodes represent alternative chemical salt forms that may affect drug absorption and bioavailability.
- **Synonym** nodes store alternative drug names across languages and naming conventions to support robust entity normalization.
- **FoodInteraction** nodes encode dietary warnings and administration guidance (e.g., "avoid grapefruit juice", "take with food").

Supplement Domain Nodes These nodes represent dietary supplements, their composition, and their therapeutic targets.

- **Supplement** nodes represent dietary supplements and include attributes such as supplement name, supplement identifier, category (vitamin, mineral, herbal, amino acid), safety rating, and clinical usage information.
- **ActiveIngredient** nodes represent chemical compounds contained in supplements that may have pharmacological activity (e.g., monacolin K in red yeast rice).
- **Symptom** nodes capture health conditions, symptoms, and wellness goals that supplements are commonly used to address (e.g., fatigue, cardiovascular health).
- **Medication** nodes represent common medications.

Nutrition and Diet Nodes These nodes support dietary deficiency analysis and nutrient-level reasoning.

- **DietaryRestriction** nodes represent common dietary patterns (e.g., Vegan, Vegetarian, Keto, Gluten-Free, Pescatarian) with descriptive metadata.
- **Nutrient** nodes represent essential vitamins, minerals, and fatty acids, including nutrient category, recommended daily allowance (RDA), and physiological roles.

1.3.2 Knowledge Graph Relationships

Relationships encode pharmacological, nutritional, and therapeutic interactions across domains. They are grouped by functional role to support interpretable multi-hop reasoning.

Drug Relationships These relationships encode pharmaceutical structure, classification, and drug–drug interactions.

- **Drug → DrugCategory (BELONGS_TO)**: Classifies drugs by therapeutic or chemical class.
- **Drug → Drug (INTERACTS_WITH)**: Represents drug–drug interactions with associated severity and mechanistic explanations.
- **BrandName → Drug (CONTAINS_DRUG)**: Links commercial brand-name products to their corresponding Drug entities.
- **Medication → Drug (MEDICATION_CONTAINS_DRUG)**: Links Myo Clinic medication entities to their underlying DrugBank Drug nodes.
- **Drug → Salt (HAS_SALT_FORM)**: Captures alternative salt formulations.
- **Drug → Synonym (KNOWN_AS)**: Supports entity resolution across naming variations.
- **ActiveIngredient → Drug (EQUIVALENT_TO)**: Identifies pharmaceutical equivalence between supplement ingredients and prescription drugs, enabling detection of hidden duplication (e.g., monacolin K and lovastatin).

Supplement Relationships These relationships encode supplement composition, therapeutic use, and adverse effects.

- **Supplement → Symptom (TREATS)**: Links supplements to conditions or wellness goals they are commonly used to address.
- **Supplement → Symptom (CAN_CAUSE)**: Captures potential adverse effects associated with supplement use.
- **Supplement → ActiveIngredient (CONTAINS)**: Connects supplements to their active chemical compounds.
- **Supplement → Medication (SUPPLEMENT_INTERACTS_WITH)**: Represents documented supplement–medication interactions, including pharmacokinetic and pharmacodynamic effects.
- **Supplement → Category (HAS_SIMILAR_EFFECT_TO)**: Connects supplements to drug categories with similar pharmacological effects, supporting detection of additive or antagonistic interactions.

Deficiency Relationships These relationships support dietary and supplement-induced nutrient deficiency reasoning.

- **DietaryRestriction → Nutrient (DEFICIENT_IN)**: Links dietary patterns to commonly deficient nutrients based on nutritional epidemiology and clinical guidelines.

- **Supplement → Nutrient (NEGATIVE_INTERACTION)**: Captures supplement-induced nutrient depletion or interference, annotated with mechanistic explanations (e.g., depletion, antagonism, masking of deficiency).
- **Drug → FoodInteraction (HAS_FOOD_INTERACTION)**: Encodes dietary restrictions and administration instructions.
- **Drug → Nutrient (INTERACTS_WITH_NUTRIENT)**: Captures drug-induced nutrient depletion or interference at the pharmaceutical level.

1.3.3 Data Sources and Integration

Table 1 summarises the three primary data sources, their entity coverage, and access methods.

Table 1: Primary Data Sources for the Knowledge Graph

Source	Coverage / Entity Types	Access Method
DrugBank	Drugs, DrugCategory, DrugProduct, Synonym, FoodInteraction, Salt; drug–drug interactions (INTERACTS_WITH) and pharmacological attributes	Licensed academic use (CSV/JSON/XML)
Mayo Clinic	Supplements, Symptom targets, Active ingredients, safety notes, contraindications, and supplement–medication interactions	Clinician-reviewed web content (structured scraping)
Curated Nutritional Data(MedlinePlus and NIH)	DietaryRestriction, Nutrient; dietary restriction–nutrient links (DEFICIENT_IN) and supplement–nutrient negative interactions (NEGATIVE_INTERACTION)	Curated from literature and clinical guidelines

Data Quality and Reliability Our data sources were selected to support clinically grounded reasoning across medication safety, supplement safety, and nutrient deficiency risk. DrugBank provides pharmacological drug information and drug–drug interactions curated from peer-reviewed pharmaceutical databases under licensed academic use. Mayo Clinic provides clinician-reviewed supplement content, including contraindications and interaction warnings reviewed by medical professionals. Dietary restriction–nutrient deficiency relationships are derived from established nutritional epidemiology research and clinical guidelines. Supplement–nutrient negative interactions are manually curated from clinical literature and stored with explicit mechanistic annotations (e.g., *depletes*, *interferes_with*, *antagonizes*, *masks_deficiency*, *competes_for_absorption*) to support transparent downstream explanations.

2 Methods

Our system uses a modular, multi-agent architecture built with LangGraph, a framework designed for stateful, graph-based AI workflows. The architecture consists of two distinct phases: a linear pre-processing pipeline that always runs first, followed by a dynamic routing loop in which a central Supervisor Agent decides which specialist to invoke next based on the patient's profile and question. All agents communicate through a shared state object, allowing intermediate results and evidence chains to persist across the full workflow.

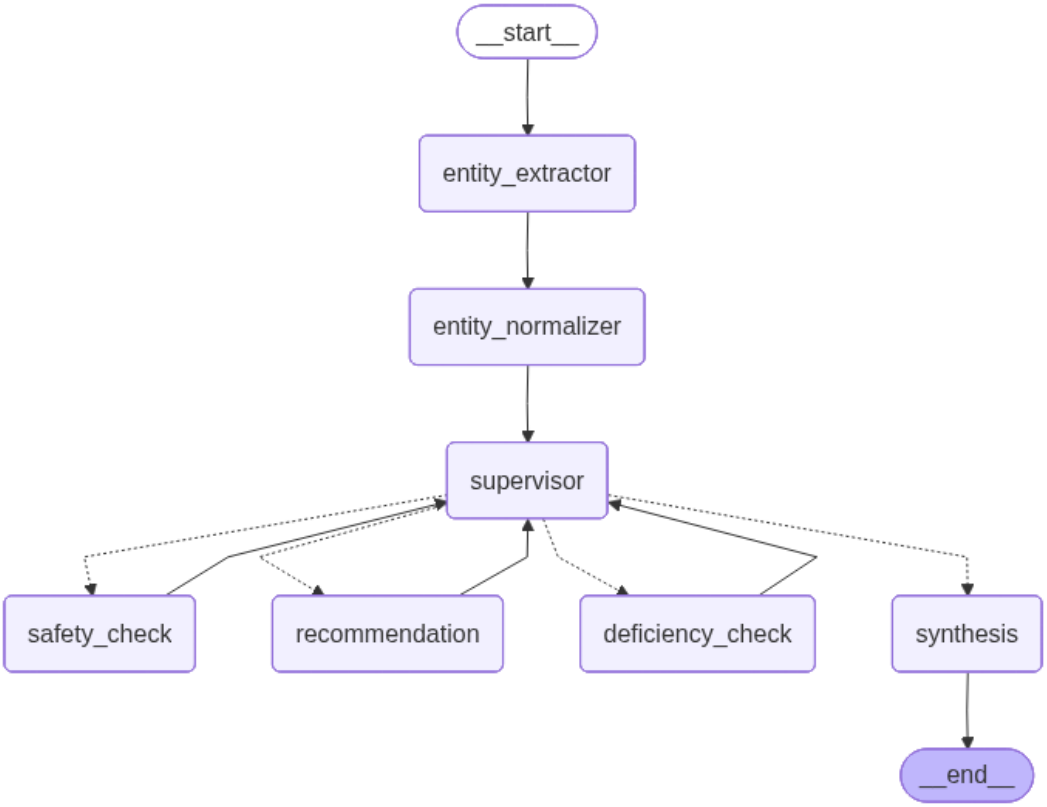


Figure 2: Sequence diagram of the workflow showing the order of tools and agents from start to end to generate a response

2.1 Agents and Tools

The system defines three component types. **Agents** are LLM-driven components that make decisions or generate queries dynamically based on context. **Hybrid** components combine an LLM call for unstructured input with deterministic parsing for structured input. **Specialist Tools** are fully deterministic, executing hardcoded Cypher queries with no LLM involvement.

Table 2: System components, their types, and roles.

Component	Type	Role
`entity_extractor`	Hybrid	LLM extraction from question; deterministic parsing for profile
`entity_normalizer`	Agent	LLM generates Cypher to map entity names to database IDs
`supervisor`	Agent	LLM routing — decides which specialist to call next
`safety_check`	Specialist Agent	LLM generates Cypher to traverse all interaction pathways
`deficiency_check`	Specialist Tool	Hardcoded Cypher — checks diet, medication, supplement pathways
`recommendation`	Specialist Tool	Hardcoded Cypher — finds supplement candidates for conditions
`synthesis`	Agent	Generates final patient-facing answer from all specialist results

Entity Extractor The Entity Extractor is a Hybrid component that runs at the start of every question and resets all state fields before processing begins, ensuring no state bleed between

questions. It extracts entities from two sources in parallel: an LLM call parses the natural language question to identify medications, supplements, conditions, and dietary restrictions — handling brand names, abbreviations, and informal language that rule-based parsers would miss; deterministic string parsing splits the structured patient profile fields into typed lists. Entities from both sources are merged and deduplicated case-insensitively, with profile entries taking priority in casing. The merged output is written to `extracted_entities` in shared state for the normalizer to consume.

Entity Normalizer The Entity Normalizer maps raw entity names to canonical database identifiers via LLM-generated Cypher queries. For each medication and supplement, it follows a three-step resolution strategy: LLM reads the full graph schema and generates a targeted Cypher query; if no results are returned, LLM generates a broader fallback query exploring alternative paths such as synonyms, brand names, and abbreviation expansions (e.g., CoQ10 → Coenzyme Q10); if the fallback also fails, the entity is recorded as `NOT_FOUND` and excluded from downstream processing. Each resolved entity is assigned a confidence level - `HIGH` (single match), `MEDIUM` (multiple matches, closest selected), or `LOW` (fallback match) - and only `HIGH` and `MEDIUM` entries populate the clean `medications_list` and `supplements_list` that downstream specialists query against. A final deduplication pass by database ID catches semantic duplicates the extractor missed (e.g., *folic acid* and *folate* to the same ID). Dietary restrictions and conditions pass through unchanged - dietary restrictions use a hardcoded lookup against the `DietaryRestriction` node, while conditions require no database mapping and are passed directly to the Supervisor for question analyses.

Supervisor The Supervisor is the central routing agent of the dynamic loop, called after entity normalization and after every specialist returns. It reads the patient profile, the user question, and compact structured summaries of specialist results accumulated in shared state, then makes a single routing decision: which specialist to call next, or whether to synthesize. After each specialist runs, the Supervisor re-evaluates the question against the accumulated specialist findings to determine whether enough information has been collected to produce a complete answer, or whether another specialist is still needed. Routing is grounded exclusively in the extracted patient data and specialist findings. It tracks which specialists have already run and will never repeat one. If all relevant specialists have run, if the question has no relevant specialists, or if no meaningful clinical picture can be inferred, the Supervisor routes directly to `synthesize`. The same fallback applies on any JSON parse error or LLM call failure, ensuring the loop always terminates cleanly. A hard cap of `MAX_ITERATIONS = 6` provides a final safety guardrail against infinite loops, forcing synthesis if the iteration limit is reached.

Recommendation The Recommendation specialist reads `conditions_list` from shared state and finds supplement candidates that treat the patient's conditions via the `TREATS` relationship in the knowledge graph. It follows a two-step query strategy: an exact case-insensitive match on symptom name first, falling back to a keyword search across individual

terms if no exact match is found. Current supplements from `supplements_list` are excluded from candidates - there is no value in recommending what the patient already takes. If no conditions are present in state, the specialist exits early and returns a `no_conditions` status ensuring the loop always terminates cleanly. Candidates are sorted by the graph's `safety_rating` property (*Generally safe* > *Use with caution* > *Not recommended*) and written to both `recommendation_results` and `candidate_supplements_list` in shared state. The complete results are written to shared state and returned to the Supervisor for re-evaluation.

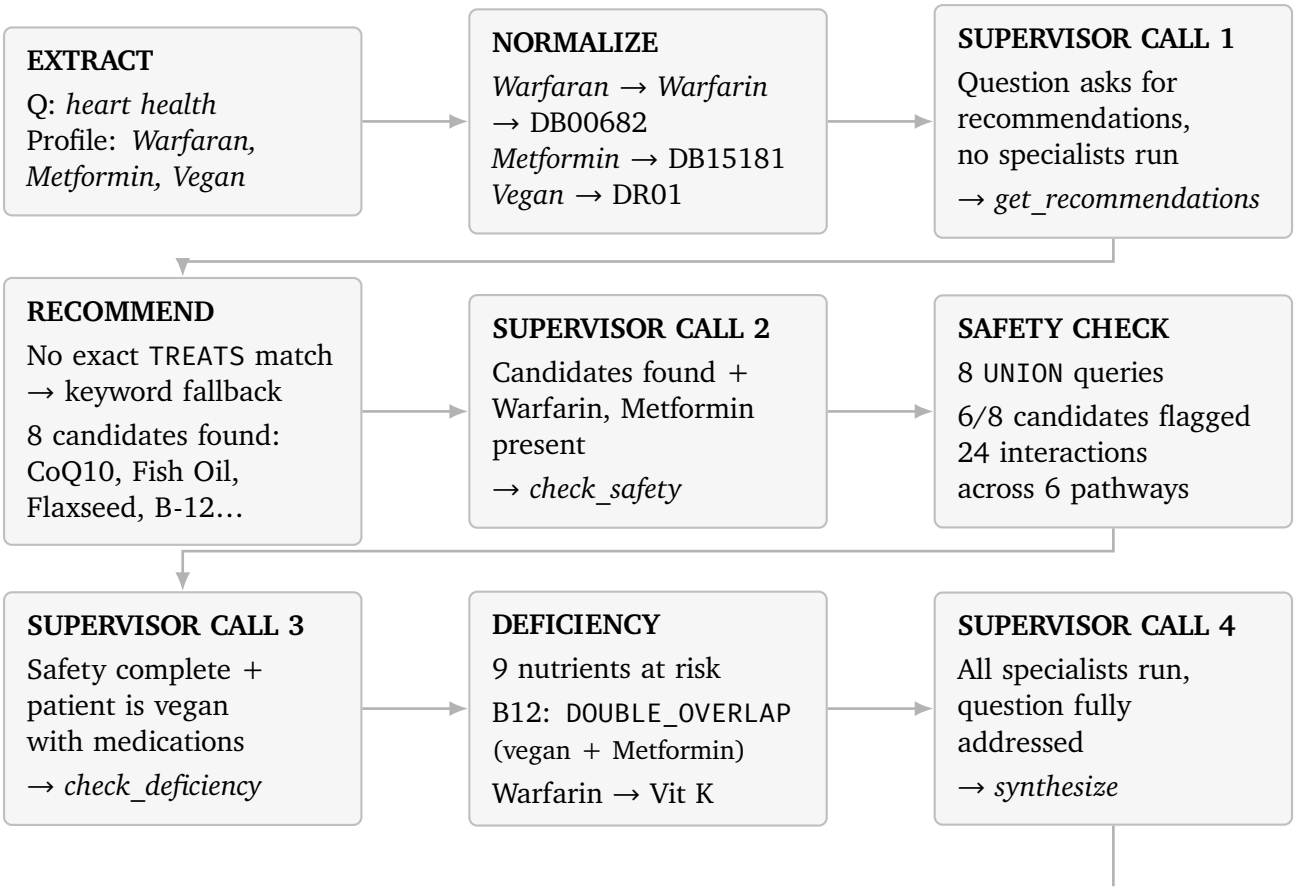
Safety Check The Safety Check specialist is the only Specialist Agent in the system - unlike the deterministic tools, it uses an LLM to generate a tailored Cypher query for each supplement. It reads `supplements_list` and `candidate_supplements_list` from shared state, deduplicates them, and checks each supplement against `medications_list`. Early exits are triggered if either list is empty, returning a `no_supplements` or `no_medications` status without querying the graph. For each supplement, the LLM receives the full graph schema and generates a `UNION` Cypher query covering all dangerous interaction paths it can identify - direct supplement–medication interactions, active ingredient equivalences, similar effect categories, and nutrient competition pathways. The query is validated via `EXPLAIN` before execution to catch syntax errors without hitting the database. If validation or execution fails, the status is explicitly set to `error` and the summary states that safety is unknown - the system never assumes safe on a query failure. The full `UNION` query is executed in a single database round-trip, and only rows that returned actual results represent flagged interactions. These are then grouped by `pathway` field and accumulated across all supplements, with per-supplement execution metadata stored in `generated_safety_queries` to support transparent evidence chains. The complete results are written to shared state and returned to the Supervisor for re-evaluation.

Deficiency Check The Deficiency Check specialist is a fully deterministic Specialist Tool that identifies nutrient deficiency risks across three independent hardcoded Cypher queries. It reads `dietary_restrictions_list`, `medications_list`, and `supplements_list` from shared state, and exits early if all three are empty. Each pathway runs independently against knowledge graph: diet pathway queries `DietaryRestriction-[DEFICIENT_IN]→ Nutrient` to identify nutrients commonly deficient under the patient's dietary pattern; the medication pathway queries `Drug -[INTERACTS_WITH_NUTRIENT]→ Nutrient` and maps `interaction_type` strings to risk levels via a lookup table (*depletes*, *antagonizes* → `HIGH`; *interferes_with_absorption*, *may_cause_loss* → `MEDIUM`; *redistributes* → `LOW`); supplement pathway queries `Supplement -[NEGATIVE_INTERACTION]→ Nutrient` and reads `severity` and `mechanism` fields directly from the graph. Results from all three pathways are merged by nutrient name. Nutrients affected by two or more sources are flagged as critical overlaps, classified as `TRIPLE_OVERLAP` (all three sources), `DOUBLE_OVERLAP` (two sources), or `SINGLE_SOURCE_MULTIPLE` (one source, multiple entries). The complete results are written to shared state and returned to the Supervisor for re-evaluation.

Synthesis The Synthesis agent is the final node in the workflow, called once the Supervisor determines that all relevant specialists have run. Before the LLM call, a pre-processing step structures the raw specialist outputs into a grounded findings block: safety interactions are grouped by supplement=medication pair with arrow-chain evidence paths (e.g., *Fish Oil → [Drug-layer] → EPA → Warfarin*); unrecognized entities are surfaced with an explicit warning that safety cannot be confirmed; if the safety status is `error`, the block states that safety is unknown and instructs the LLM not to assume safe; deficiency findings are grouped by nutrient with critical overlap flags; and recommendation candidates are flagged if they also appear in the safety interactions. This structured findings block, together with the full evidence chain accumulated across all specialist calls, is passed to the LLM with explicit instructions to ground its response strictly in the specialist findings - no external biomedical knowledge - and to present the answer in plain, accessible language as a doctor would explain findings to a patient. If the LLM call fails or no API key is present, a deterministic fallback generates a structured markdown summary directly from state without any LLM call, ensuring the workflow always produces a final answer.

2.1.1 End-to-End Example

User Question: *"What supplements are good for heart health?"*
Patient Profile: Medications *Warfaran, Metformin*; Dietary Restriction *Vegan*.



continued from previous page (Supervisor 4 → synthesize)

SYNTHESIS

Given your medications, most heart supplements will interact dangerously with your Warfarin and Metformin, so I cannot recommend the typical options.

Coenzyme Q10 interacts with both your medications through multiple pathways. It contains compounds equivalent to drugs that directly interact with your Warfarin and Metformin, plus it has anticoagulant effects similar to your Warfarin, which could increase bleeding risk.

Fish oil also creates problems with both medications. The EPA and DHA it contains are equivalent to prescription drugs that interact with your Warfarin and Metformin. Fish oil also has blood-thinning effects that could amplify your Warfarin's action.

Folate and **Vitamin B-12** both interact directly with your medications through active ingredients that are pharmaceutical equivalents. B-12 has a direct interaction with your Metformin, while folate affects the same vitamin B-12 pathway that your Metformin already disrupts.

Flaxseed oil has a moderate interaction with your Warfarin by causing gas, which can affect medication absorption. **Vitamin B-6** and **Vitamin A** also interact with both your medications through their active ingredients.

How we found this: Coenzyme Q10 → contains → Ubiquinone → equivalent to → Ubidecarenone → interacts with → Warfarin

See Appendix A for a detailed discussion of design choices and trade-offs.

3 Results

3.1 Evaluation Methodology

We evaluated the system using a golden-set approach: 62 test cases with known correct answers, each derived from verified relationships in the knowledge graph confirmed via direct Cypher queries. Every test was executed through the complete multi-agent pipeline and the final synthesized response was manually labeled as *Correct*, *Partially Correct*, or *Incorrect*.

Tests were organized into four suites, each targeting a different system component:

- **Safety Check (22 tests):** Whether the system detects known dangerous supplement-medication interactions and correctly reports safe pairs as safe. Tests range from missing inputs and unknown entities to multi-supplement × multi-medication interaction matrices.
- **Deficiency Check (12 tests):** Whether the system identifies nutrient risks across all three pathways (diet-based, medication-induced, supplement-induced) and detects compounded risk when multiple sources affect the same nutrient.
- **Recommendation (15 tests):** Whether the system retrieves relevant supplement candidates for health conditions, excludes supplements the user already takes, and

handles conditions with limited knowledge graph coverage gracefully.

- **Multi-Agent End-to-End (13 tests):** Whether the supervisor correctly routes to multiple specialists, whether the recommendation-to-safety pipeline catches dangerous candidates, and whether the system handles maximum-complexity inputs without looping or crashing.

In addition to the specialist-level suites, the test cases collectively exercise the pre-processing pipeline components that all agents depend on:

- **Question classification:** Ambiguous and open-ended question types were included across all suites to test whether the LLM correctly classifies intent. For example, "I take Carbamazepine and follow a vegan diet. Any concerns?" does not explicitly request safety or deficiency analysis, requiring the supervisor to infer that both are relevant.
- **Entity extraction:** Tests with missing question types (e.g., no supplements mentioned, no medications provided) and missing database entities (e.g., fabricated condition names, supplements not in the knowledge graph) were used to verify that the entity extractor correctly handles incomplete or unrecognizable inputs without crashing or hallucinating entities.
- **Entity normalization:** Extracted results were checked to confirm correct matching against database identifiers, including resolution through brand names, synonyms, and the LLM-based typo correction fallback. Tests verified that normalized entity lists align with canonical knowledge graph entries before being passed to downstream agents.

3.2 Results Summary

Table 3: Golden-set evaluation results across all test suites.

Test Suite	Tests	Correct	Partially Correct	Accuracy
Safety Check	22	17	5	77%
Deficiency Check	12	12	0	100%
Recommendation	15	10	5	67%
Multi-Agent (E2E)	13	11	2	85%
Overall	**62**	**50**	**12**	**81%**

The system produced zero incorrect responses across all 62 tests. No unsafe recommendations were generated, and all 12 partially correct results stem from knowledge graph coverage gaps, data quality issues, or minor LLM imprecision rather than flawed reasoning or hallucination. Three false positives were identified, each traceable to a specific knowledge graph data quality issue. One false negative was identified: 14 of 15 known dangerous supplement-medication interactions were correctly detected, with the single miss caused by a missing graph edge rather than a reasoning failure.

3.3 What the System Gets Right

Safety detection The system correctly flagged 14 of 15 known dangerous interactions in the knowledge graph. For multi-entity inputs (e.g., two supplements checked against two medications), it produced the complete interaction matrix without missing any pairwise combination. The single missed interaction was caused by a missing edge in the knowledge graph, not a failure in the system's reasoning or query logic.

Deficiency detection All three pathways performed correctly: diet-based deficiencies returned accurate nutrient counts for all six dietary restrictions in the database, medication-induced depletions were identified with correct mechanisms and severity, and supplement-induced depletions surfaced the expected nutrient interference. The cross-pathway overlap detection, where the same nutrient is affected by multiple independent sources, worked as designed, including a triple overlap on Calcium from a vegan diet, Carbamazepine, and Zinc supplementation simultaneously.

Recommendation-to-safety pipeline When the recommendation agent surfaced supplement candidates for a health condition, the supervisor automatically routed to the safety agent to check those candidates against the user's medications. In cases where a recommended supplement was dangerous (e.g., Aloe recommended for constipation but contraindicated with Warfarin), the synthesized response correctly warned the user and identified safe alternatives.

Supervisor routing The supervisor made correct routing decisions in the majority of multi-agent tests, completing all workflows within 2 to 4 iterations without hitting the maximum iteration cap or entering infinite loops.

3.4 What the System Gets Wrong

The 12 partially correct results fall into four categories. For each, we walk through a representative example to illustrate the failure mode.

1. **Knowledge graph coverage gaps (5 cases):** Common conditions users are likely to ask about, such as joint pain, fatigue, headaches, inflammation, and general sleep issues, lack `TREATS` relationships in the current graph. The system handled these gracefully but could not provide useful supplement candidates.
   Example: When asked "What supplements can help with my joint pain?", the system found no exact match for "Joint Pain" as a symptom node. The keyword fallback split the query into ["joint", "pain"] and matched on "pain" broadly, returning Evening Primrose (indicated for breast pain) and Marijuana (indicated for general pain). Neither is a joint-specific recommendation, and the obvious candidate, Glucosamine, was absent because it lacks a `TREATS` edge to any joint-related symptom in the knowledge graph.

2. **Knowledge graph data quality (3 cases):** An incorrect `EQUIVALENT_TO` edge maps DHA (an omega-3 fatty acid in fish oil) to Bupropion (an antidepressant), producing a spurious interaction pathway and contributing to the identified false positives. Additionally, the system lacks supplement-supplement interaction detection, so it cannot verify whether two supplements are safe to take together when no medication is involved.
   Example: When checking Fish Oil against Warfarin, the system correctly detected the real interaction via EPA equivalence to Icosapent, but also reported a second pathway: DHA → equivalent to → Bupropion → interacts with → Warfarin. This pathway is pharmacologically incorrect since DHA is not equivalent to Bupropion, but it appeared because the `EQUIVALENT_TO` edge exists in the graph. The real interaction was still detected, but the false pathway inflates the reported interaction count.
3. **LLM query imprecision (2 cases):** The LLM-generated safety Cypher occasionally includes speculative `CAN_CAUSE` → `Symptom` pathways that produce false positive interactions. The entity normalizer also mapped "Aspirin" to "Nitroaspirin," a related but distinct compound. Both issues are conservative errors that flag potential risks rather than miss real dangers.
   Example: When checking Zinc against Warfarin (a pair with no known direct interaction), the LLM-generated Cypher included a pathway through `CAN_CAUSE` edges linking Zinc's side effects to symptoms that also appear in Warfarin's drug category. This produced 8 speculative "Symptom-Effect" interactions at MODERATE severity, none of which reflect a real pharmacological interaction between Zinc and Warfarin.
4. **Supervisor routing (2 cases):** The supervisor occasionally synthesized after completing one specialist even when the patient profile contained data relevant to additional agents. This typically occurred when a dangerous interaction was already found and the supervisor judged the safety finding sufficient to answer the user's question, skipping deficiency analysis even though the patient's dietary restrictions warranted it. Importantly, this early-stop behavior does not produce unsafe outputs since the safety-critical analysis was completed; rather, it results in an incomplete but not incorrect response.
   Example: A patient taking Carbamazepine with a vegan diet asked "Any concerns?" The supervisor routed to the deficiency agent, which correctly identified 11 nutrients at risk with 3 critical overlaps between the vegan diet and Carbamazepine depletions. However, the supervisor then synthesized without running the safety agent, even though Carbamazepine is a medication that could interact with supplements. No supplements were in the profile so no interaction existed, but the safety check was still warranted as a precautionary measure.

None of these issues produced unsafe outputs. The failure modes are either conservative (false positives, incomplete coverage) or cosmetic (noisy entity extraction), and each maps directly to a concrete improvement target.

4 Discussion

The evaluation results confirm that the system's core architecture works as intended: a supervisor agent can dynamically route across specialized tools, aggregate their findings, and produce coherent responses grounded in explicit knowledge graph evidence. The strongest performance came from the components with the most structured data behind them, as deficiency detection had curated, deterministic graph relationships, while recommendation accuracy was lowest because it depends on symptom coverage that the knowledge graph does not yet fully provide. This pattern suggests that expanding the data layer will yield larger accuracy gains than further refining the agentic logic.

The safety results are encouraging but imperfect. The multi-pathway query design detected the vast majority of known dangerous interactions with no unsafe recommendations, but the single false negative, caused by a missing graph edge, illustrates a fundamental constraint of any knowledge-graph-grounded system: it can only reason over relationships that exist in the data. Similarly, the false positives traced to an incorrect `EQUIVALENT_TO` edge show that data quality directly determines output quality, regardless of how well the agents reason.

The recommendation-to-safety pipeline is arguably the system's most valuable capability. No existing consumer-facing supplement tool recommends a supplement and then automatically checks it against the user's medications before presenting it. This pipeline caught dangerous candidates in every test where one existed, and the fact that it operates without any user intervention makes it a meaningful improvement over manual interaction checking.

4.1 Limitations and Future Work

Knowledge graph coverage The most impactful limitation is the scope of the knowledge graph. The current graph covers approximately 28 supplements from Mayo Clinic and a curated set of drug interactions from DrugBank, which is sufficient to demonstrate the architecture but far from comprehensive. Common conditions that real users ask about, such as joint pain, fatigue, headaches, inflammation, and sleep issues, lack `TREATS` relationships entirely. Expanding the graph to include broader symptom-supplement mappings from sources like the Natural Medicines Database or Examine.com would directly address these gaps.

Knowledge graph data quality and external validation The incorrect DHA-Bupropion equivalence edge and the Aspirin-Nitroaspirin normalization error demonstrate that even small data quality issues propagate through multi-hop reasoning chains and produce visible output errors. Because the knowledge graph is a static snapshot, it cannot reflect newly discovered interactions or updated clinical guidelines until manually rebuilt. Future work should include automated validation scripts that cross-reference `EQUIVALENT_TO` edges against pharmacological databases, as well as integration with external biomedical databases such as PubMed for up-to-date interaction and safety data. Clinical validation

through pharmacist or physician review of system outputs would further assess whether the recommendations, warnings, and deficiency analyses are clinically appropriate for real-world use.

Supplement-supplement interactions The current system only detects supplement-medication interactions. It cannot assess whether two supplements are safe to take together, which is a common and clinically relevant user question. Adding `Supplement` → `INTERACTS_WITH` → `Supplement` edges to the knowledge graph and extending the safety agent to query them would address this gap.

Deficiency to recommendation enrichment Currently, the recommendation agent queries only by health condition using `conditions_list`. It does not use identified nutrient deficiencies to find supplements that could address those gaps. For example, if the deficiency agent identifies that Metformin depletes Vitamin B12, the recommendation agent has no mechanism to suggest a B12 supplement based on that finding. The synthesis agent partially bridges this gap through LLM reasoning over the combined results, since most nutrients have well-known corresponding supplements. However, this relies on the LLM's general knowledge rather than structured graph data, and name-based inference is unreliable (e.g., "Fish Oil" does not contain "Omega-3" in its supplement name). Adding a `Supplement` → `REPLENISHES` → `Nutrient` relationship to the knowledge graph would enable data-driven deficiency-aware recommendations without depending on LLM reasoning for this connection.

Supervisor routing The supervisor occasionally stops after a single specialist even when the patient profile contains data relevant to additional agents. This typically occurs when a high-severity safety finding is detected early, and the supervisor judges it sufficient to answer the user's question without checking other dimensions. While this early-stop behavior does not produce unsafe outputs, it can result in incomplete analyses. Improving the supervisor's completion criteria to check whether all populated profile fields have been addressed, rather than relying solely on whether the immediate question has been answered, would reduce these cases.

LLM reliability The system depends on Claude for entity extraction, entity normalization, Cypher query generation, supervisor reasoning, and response synthesis. Each of these steps introduces a risk of hallucination or imprecision. The speculative `CAN_CAUSE` pathways in safety queries and the entity extraction noise observed in testing are mild examples, but more adversarial or ambiguous inputs could produce more significant errors. Replacing LLM-generated Cypher with validated query templates for critical safety paths, or adding a post-generation validation step, would reduce this risk.

5 Conclusion

This project addresses a critical gap in consumer health guidance: the absence of personalized, evidence-grounded supplement safety analysis. Despite the supplement industry being so prominent in people's diets with minimal regulation, no existing tool analyzes a user's complete health picture to identify dangerous interactions, hidden pharmaceutical overlaps, and medication-induced deficiencies simultaneously.

We built a system that combines a Neo4j biomedical knowledge graph with a multi-agent LangGraph workflow powered by Claude. Given a patient profile and user question, a supervisor agent dynamically routes across specialized tools for safety checking, deficiency detection, and personalized recommendation. All findings are aggregated by a synthesis agent into a plain-English response backed with explicit knowledge graph evidence. The hybrid architecture combines deterministic graph traversal for critical safety checks with agentic LLM reasoning for flexible query interpretation and response generation.

Evaluation on 62 golden-set test cases showed that the system produces reliable outputs, with no unsafe recommendations generated across the full test suite. The system correctly detected the vast majority of known dangerous interactions, identified nutrient deficiencies across all three pathways including compounded cross-pathway risks, and demonstrated the ability to recommend supplements and then automatically catch dangerous candidates before presenting them to the user. The partially correct results traced entirely to knowledge graph coverage gaps and data quality issues rather than reasoning failures, confirming that the architecture is sound and that future accuracy gains will come primarily from expanding and refining the underlying data.

Future work will focus on broadening knowledge graph coverage to include more symptom-supplement relationships, auditing equivalence edges to eliminate false interaction pathways, adding supplement-supplement interaction detection, and validating outputs against clinical expert review. These improvements would move the project toward a more complete supplement advisory system that helps people make safer, more informed decisions about their health.

References

Huang, Zhen, Yang Liu, Yuchen Zhang, Yingxia Wang, Qiang Chen, Wei Zhou, and Hao Wu. 2024. "Biomni: A Unified Framework for Biomedical Multi-Task Learning with Large Language Models." *arXiv preprint arXiv:2402.10391*

Qu, Kaixuan, Yuhang Liu, Jingyu Guo, Yuyang Wang, Zeyuan Chen, Jianfeng Tang, and Wei Wang. 2024. "CRISPR-GPT: An LLM Agent for Automated Design of Gene-Editing Experiments." *bioRxiv*

Yao, Shunyu, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. "ReAct: Synergizing Reasoning and Acting in Language Models." *arXiv preprint arXiv:2210.03629*

Appendices

A Workflow Design Considerations

This section summarizes key benefits and trade-offs of the multi-agent workflow architecture used in the system.

A.1 Design Choices

This section summarizes the key architectural decisions made during system design and the rationale behind each.

LangGraph over a Single LLM Call

A single LLM call could theoretically answer supplement queries directly from parametric knowledge, but this approach offers no safety guarantees, no auditability, and no mechanism to verify claims against a curated knowledge base. LangGraph's multi-agent structure instead grounds every finding in explicit graph queries, with each specialist operating independently against the Neo4j database. This modularity allowed each component to be developed, tested, and evaluated in isolation - demonstrated by the per-agent test runners and golden datasets built for each specialist. Shared state allows findings from one specialist to inform the next, most critically the safety check consuming `candidate_supplements_list` written by the recommendation specialist.LangGraph's native integration with LangSmith provides full execution traces - every node invocation, routing decision, and state transition is logged and inspectable, making it straightforward to diagnose failures, trace how a final answer was reached, and verify that the correct specialists ran in the correct order for any given query.

Supervisor Routing Loop over a Fixed Pipeline

A fixed pipeline would run all three specialists on every query regardless of relevance. The Supervisor routing loop avoids this by re-evaluating the question after each specialist returns and routing only to specialists whose findings are needed to answer the question. A pure safety question skips recommendation entirely; a question with no medications skips safety check. This reduces unnecessary LLM and database calls while ensuring that when specialists are sequenced - such as routing to safety after recommendation - the ordering is driven by what the question actually requires rather than a hardcoded execution order.

Neo4j Knowledge Graph over a Flat Database or Pure LLM Reasoning

Supplement–medication interactions are inherently relational - a supplement may interact with a medication not directly but through a shared active ingredient, a common pharmacological category, or a nutrient both affect. A flat database cannot express these multi-hop paths without expensive joins, and a pure LLM would hallucinate or miss interactions not present in its training data. Neo4j's graph structure naturally models these traversal paths, and using an explicit knowledge base means the system's reasoning is fully traceable to specific nodes and relationships. Critically, separating knowledge from reasoning means the graph can be corrected and extended - adding a missing interaction edge or fixing a bad relationship - without modifying any agent logic. Data quality issues identified during evaluation, such as the spurious `DHA` → `EQUIVALENT_TO` → `Bupropion` relationship generating false positives, could be pinpointed to specific graph edges and resolved at the source rather than requiring changes to agent behaviour.

A.2 Benefits

The workflow uses a supervisor-based structure in which specialized agents perform focused analyses and contribute results to a shared state before a final synthesis step generates the response. This design provides several advantages:

- **Flexible multi-step reasoning.** The system can dynamically combine multiple analyses such as recommendation generation, safety validation, and deficiency detection within a single query.
- **Modular architecture.** Individual agents operate independently, allowing new analytical components to be added without significant changes to the overall workflow.
- **Shared intermediate context.** Agent outputs are stored in a shared state, allowing later steps to incorporate earlier findings when generating the final response.
- **Traceability and auditability.** Every node invocation, routing decision, and state transition is logged via LangSmith, making it straightforward to inspect how a final answer was reached and verify that the correct specialists ran for any given query. In a safety-critical domain, this auditability is essential - every interaction flag and deficiency finding is traceable to a specific graph relationship rather than opaque model reasoning.

A.3 Trade-offs

While the workflow improves flexibility and extensibility, it also introduces several trade-offs:

- **Increased latency.** Queries may require multiple agent invocations, which can increase response time compared to single-step systems.
- **Routing variability.** Because the supervisor determines which agents to invoke, workflow paths may vary across similar queries.
- **Data quality dependency.** The system's accuracy is directly tied to the completeness and correctness of the knowledge graph. Missing edges produce false negatives - as seen with the absent St. John's Wort and Sertraline interaction — while incorrect relationships produce false positives, as demonstrated by the spurious `DHA` → `EQUIVALENT_TO` → `Bupropion` edge generating artificial interactions during evaluation. Agent logic cannot compensate for gaps or errors at the data layer.