Bilkent University
Department of Computer Engineering

Senior Design Project
T2526
Consensia

Analysis and Requirement Report

22001482, Ahmed Haikal, ahmed.haikal@ug.bilkent.edu.tr
22003421, Hakan Karakoç, hakan.karakoc@ug.bilkent.edu.tr
22101535, Amirhossein Ahani, amirhossein.ahani@ug.bilkent.edu.tr
22102331, Türker Köken, turker.koken@ug.bilkent.edu.tr
22002119, Mehmet Hakan Yavuz, mehmethakanyavuz02@gmail.com

Anil Koyuncu

İlker Burak Kurt
Mert Bıçakçı

22-11-2025

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS491/2.

Contents

Analysis and Requirement Report

Consensia: llm as a judge

1 Introduction

This report presents the analysis and requirements of **Consensia**, a prototype system designed to study how large language models (LLMs) can be used to generate judgments and reach consensus across multiple AI-generated perspectives. The primary objective of the project is to explore the feasibility, limitations, and reliability of LLM-based decision-making mechanisms rather than to provide authoritative or fully autonomous decisions.

Consensia allows users to define multiple AI personas representing different professional roles commonly found in software engineering and business contexts, such as Chief Technology Officer (CTO), Software Architect, Senior Developer, Quality Assurance (QA) Engineer, Product Manager, and Finance/CFO. Each persona independently generates a response to a given technical or strategic question based on its predefined role characteristics.

After persona responses are generated, a separate large language model, referred to as the **Judge LLM**, analyzes the collected answers. The Judge LLM evaluates the responses according to predefined criteria such as consistency, fairness, and clarity of reasoning, and produces a consolidated judgment accompanied by an explanation of the decision-making process. When human-labeled data is available, the Judge LLM's output can be compared against ground-truth answers in order to assess alignment and reliability.

The system is intended as an experimental and educational platform that supports research into multi-agent LLM behavior, explainable AI, and trust in AI-assisted decision support systems. Consensia does not aim to replace human expertise and should be considered a recommendation and analysis tool rather than a definitive authority in real-world decision-making scenarios.

2 Current System

Consensia is developed as a largely greenfield project; however, there are existing practices, tools, and research efforts that partially address aspects of the problem it aims to explore. These approaches do not provide a complete or integrated solution but represent the current state of how similar problems are typically handled.

A widely used approach in practice is querying a single large language model, such as OpenAI's GPT-based models or Google's Gemini models, to obtain recommendations or explanations for technical or strategic questions [2], [3]. While these models are highly capable, they generate responses from a single perspective and do not explicitly model disagreement, role-based reasoning, or structured evaluation. As a result, users must rely on their own judgment to assess response quality, correctness, and potential bias.

Another common workaround involves manually prompting a language model to role-play multiple personas within a single interaction. In such cases, users request the model to simulate different professional roles (e.g., developer, manager, or architect) and then subjectively compare the generated responses. This process is informal, lacks reproducibility, and does not provide systematic evaluation, justification, or traceability of the final decision. From a software engineering perspective, such ad-hoc solutions do not follow structured analysis or modeling principles as discussed in classical object-oriented system design methodologies [1].

In research contexts, recent studies have explored multi-agent and persona-based language model systems, where multiple LLM instances generate diverse responses and interact through debate or coordination mechanisms [5], [6], [10]. These works demonstrate that multi-agent setups can improve reasoning quality and expose alternative viewpoints. However, these approaches are typically implemented as experimental pipelines and are not presented as user-facing systems that support configurable persona definitions, transparent judgment, and optional comparison against human-labeled ground truth.

To the best of our knowledge, there is no existing platform that integrates role-based AI persona generation, independent response production, explicit LLM-based judgment, and explainable evaluation within a unified system. Consensia aims to address this gap by providing a structured experimental platform for studying how large language models can act as judges, how their reasoning can be analyzed, and how their judgments compare with human-labeled data.

3 Proposed System

3.1 Overview

Consensia is a multi-agent LLM evaluation platform designed to analyze how large language models produce judgments and reach consensus across diverse AI personas. Users can generate multiple AI agents — each with different technical or managerial roles such as CTO, Senior Developer, QA Engineer, SRE, or Product Manager — and prompt them with a software engineering or strategic question.

Each persona responds based on its predefined role characteristics. A dedicated Judge LLM then evaluates all persona answers using criteria such as:

- Consistency
- Fairness
- Reasoning clarity
- Majority voting/plurality consensus
- Alignment with human-ground truth (when available)

The final output includes:

- A computed consensus
- A textual explanation
- A justification of why certain persona answers were more reliable
- A confidence score

This system can later be extended to function as a research tool for AI ethics, decision support, trust metrics, and educational use.

3.2 Functional Requirements

1 — Persona Creation

- The system shall allow users to create multiple AI personas with predefined roles.
- The system shall allow users to define custom attributes, including expertise level.
- The system shall support persona creation from structured inputs such as textual descriptions or uploaded CVs.

2 — Persona Response Generation

- For each input question, the system shall generate an independent answer from every persona.
- Persona responses shall be generated without access to other personas' answers to ensure independence.
- The system shall store all persona-generated responses for later analysis and evaluation.

3 — Judge LLM Analysis

- The system shall collect all personas answers and send them to the Judge LLM.
- The Judge LLM evaluation shall include:
    - consistency
    - fairness
    - clarity of reasoning

    - correctness (when a ground-truth label exists)

4 — Judge Based Final decision

- The system shall designate a single LLM instance as the Judge.
- The Judge LLM shall evaluate all persona responses holistically without applying explicit voting or aggregation algorithms.
- The Judge LLM shall determine a final judged answer based on its internal reasoning and evaluation criteria.
- The Judge LLM's decision shall be presented as a recommendation rather than an objective or guaranteed-correct result.

5 — Explanation Generation

- The Judge LLM shall generate a written explanation that includes:
    - justification for the selected consensus answer
    - discussion of conflicting persona answers
    - description of reasoning criteria used

6 — Result Comparison

- When human-labeled data is available, the system shall compare the Judge LLM output with ground truth.
- The system shall compute evaluation metrics such as reliability or accuracy score.
-

7 — User Interface Interaction

- The system shall allow users to:
    - Create personas
    - Enter questions
    - View persona answers
    - View the judge's consensus and explanation

8 — Data Storage

- The system shall store:
    - persona definitions
    - persona-generated answers
      judge outputs
    - evaluation metrics

3.3 Non-functional Requirements

1 — Performance

- Judge LLM evaluation shall respond within **5–10 seconds** under normal conditions.
- Persona answers should be produced within **3–7 seconds,** depending on the provider.

2 — Reliability

- The system shall retry failed LLM calls up to 3 times.
- The system must handle rate limits gracefully.

3 — Security

- API keys must be stored securely in environment variables.
- The system shall enforce CORS and only allow trusted origins.

4 — Scalability

- The system must support the addition of new LLM providers (Ollama, OpenAI, Gemini, Claude).

5 — Maintainability

- Codebase shall follow a modular architecture with separate services for:
    - Persona generation
    - Judge analysis

- Storage
- UI interactions

6 — Usability

- UI must be simple, clean, and intuitive for non-technical users.

3.4 Pseudo Requirements

1) The UI should provide light/dark themes.

3) The system may include role templates for quick persona creation.

4) The Judge LLM should support Gemini models.

3.5 System Models

3.5.1 Scenarios

Scenario 1 — Basic Persona Query

Goal: User asks a question to multiple personas and receives a Judge LLM evaluation.

1. The user opens the Consensia interface.
2. The user creates three personas: CTO, Senior Developer, and QA Engineer.
3. The user enters a question: “Which architecture should we use for scaling our backend?”
4. The system generates three independent persona answers.
5. Judge LLM analyzes all persona answers (no voting; pure judgment).
6. Judge LLM produces:
    - a final recommended answer
    - a detailed explanation
    - reasoning scores (consistency, fairness, clarity)
7. The system displays all persona answers + judge consensus.

Scenario 2 — Persona from CV

Goal: User generates a persona using a CV.

1. The user clicks “Create Persona from CV.”
2. User uploads or pastes a CV/resume.
3. System extracts:
    - Title (e.g., “Senior Software Engineer”)
    - Experience summary
4. The system generates a structured persona profile.
5. A person is saved and can answer questions like any other.

Scenario 3 — Ground-Truth Evaluation (Optional)

Goal: Judge LLM's answer is compared to human-labeled datasets.

1. The user selects a benchmark question with a known human-labeled answer.
2. Personas respond independently.
3. Judge LLM evaluates the persona answers and outputs the final judged answer.
4. The system compares the judge's decision with the ground-truth answer.
5. The system displays:
    - Correct/incorrect
    - Reliability score
    - Notes on reasoning quality

3.5.2 Use-Case Model

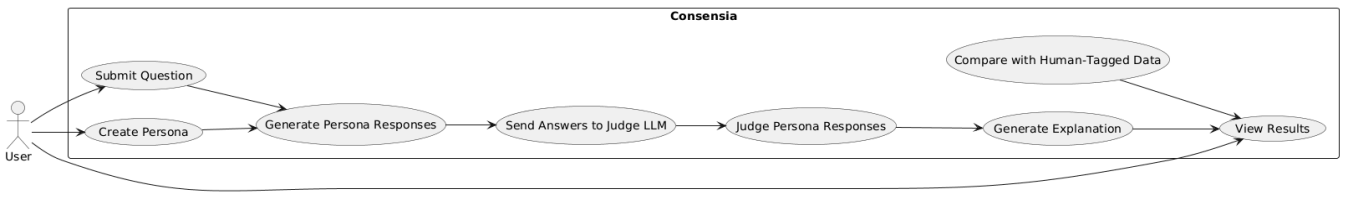

Figure 1: Use-Case Model Diagram

3.5.3 Object and Class Model

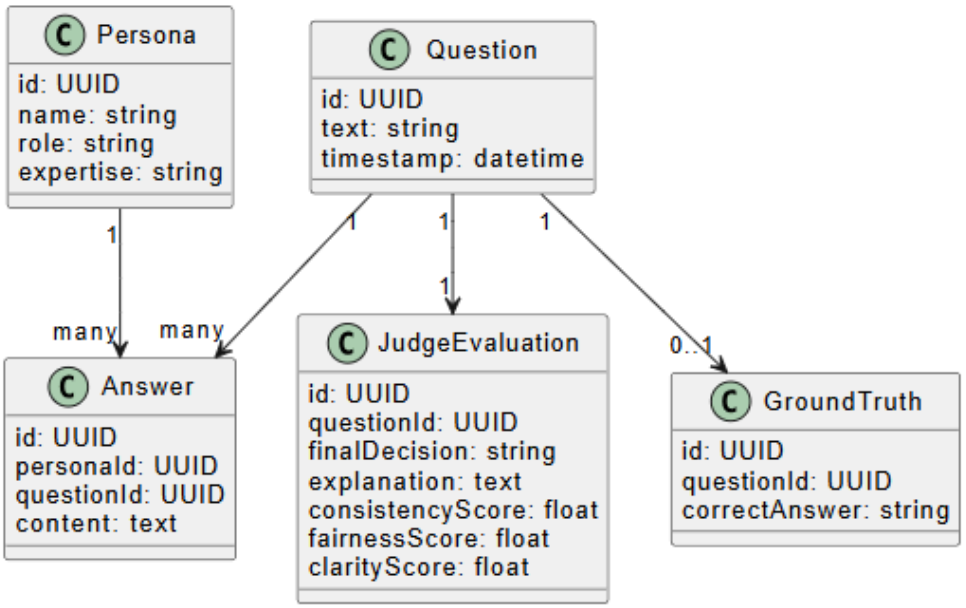

Figure 2: Object and Class Model

3.5.4 Dynamic Models

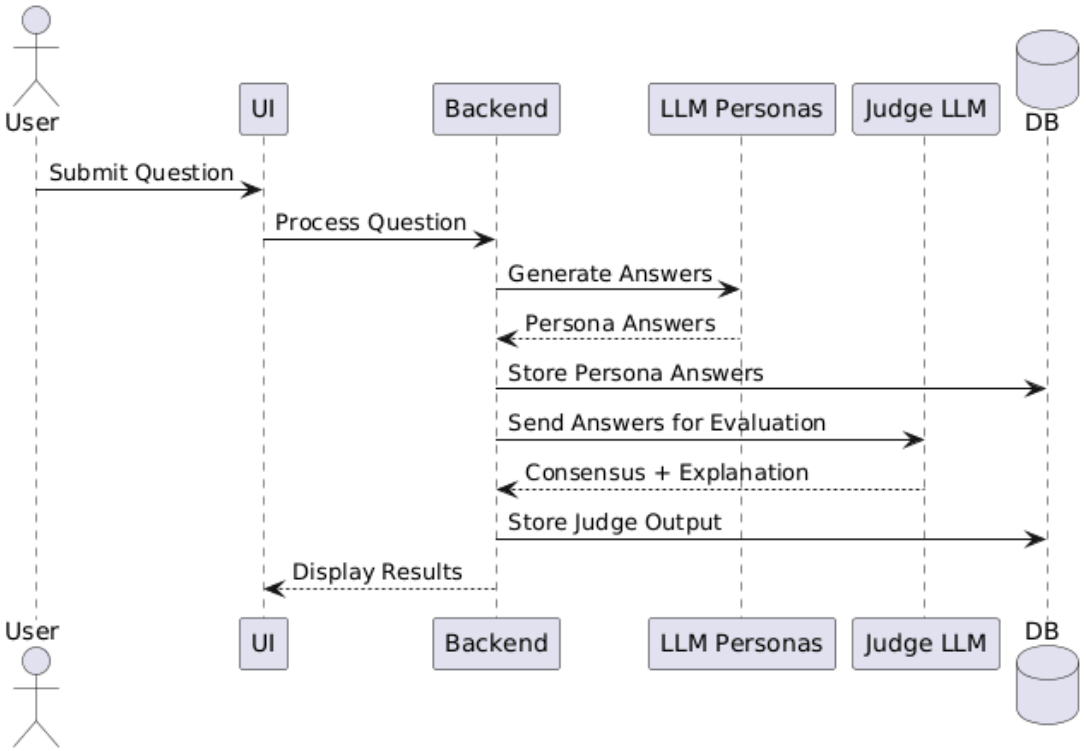

Figure 3: Sequence Diagram

3.5.5 User Interface

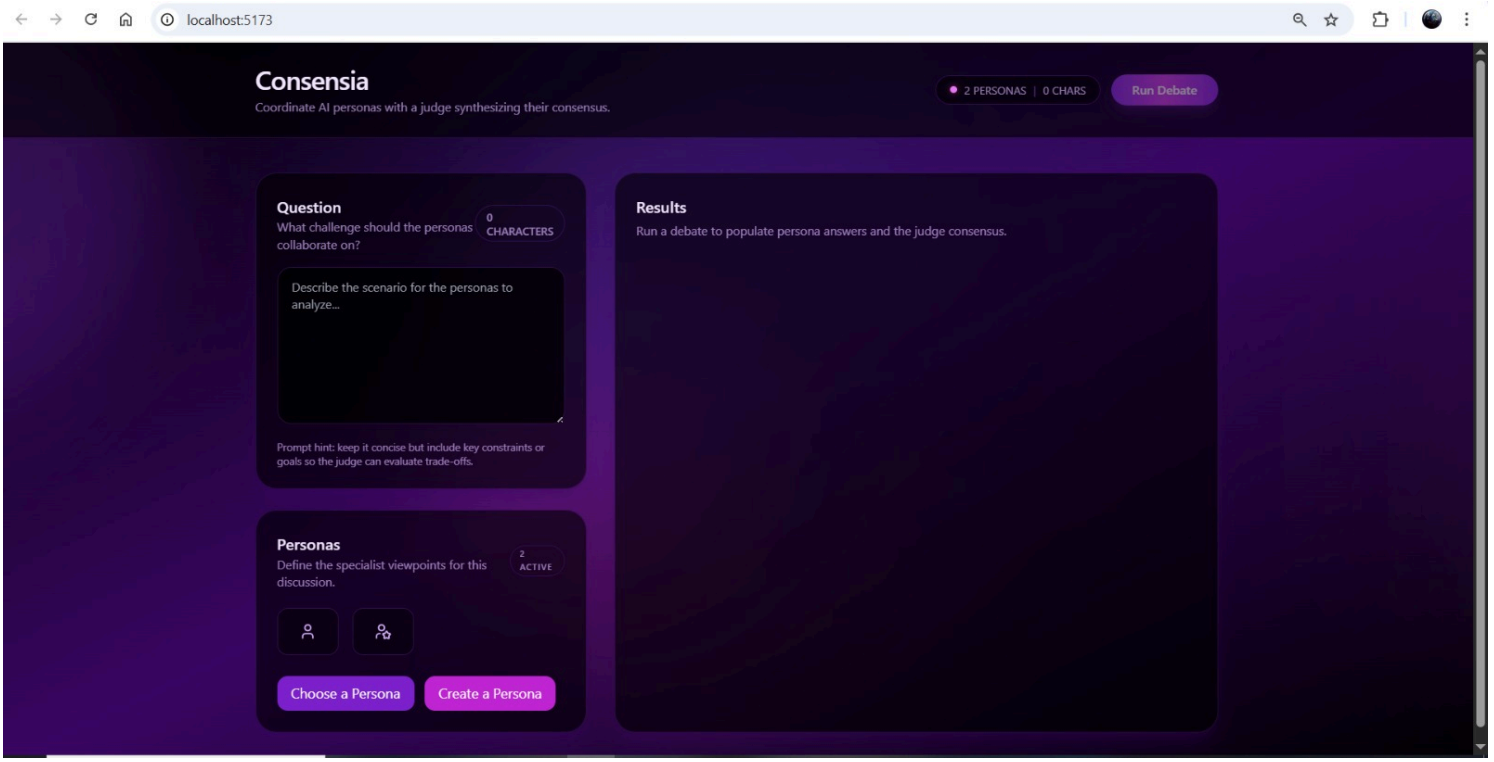

Figure 4: HomePage UI

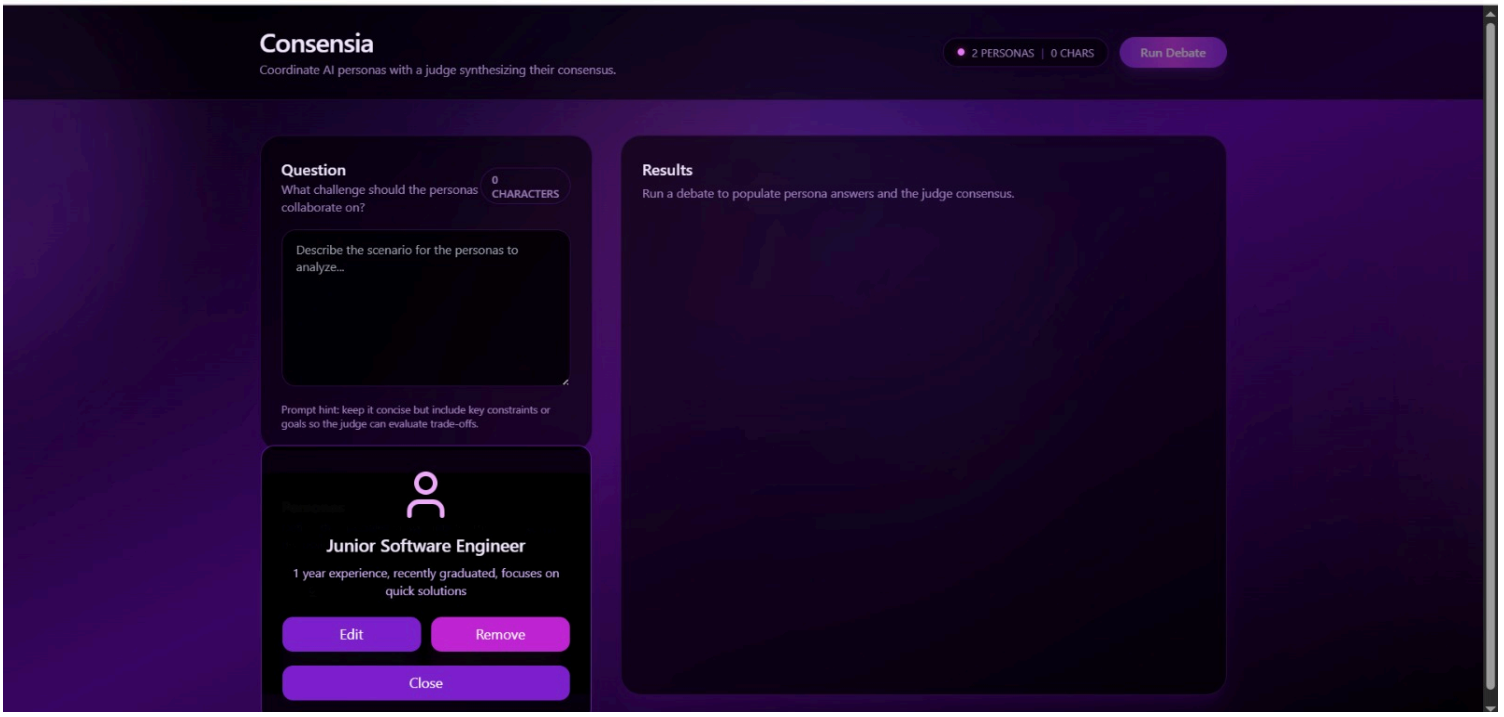

Figure 5: Persona view

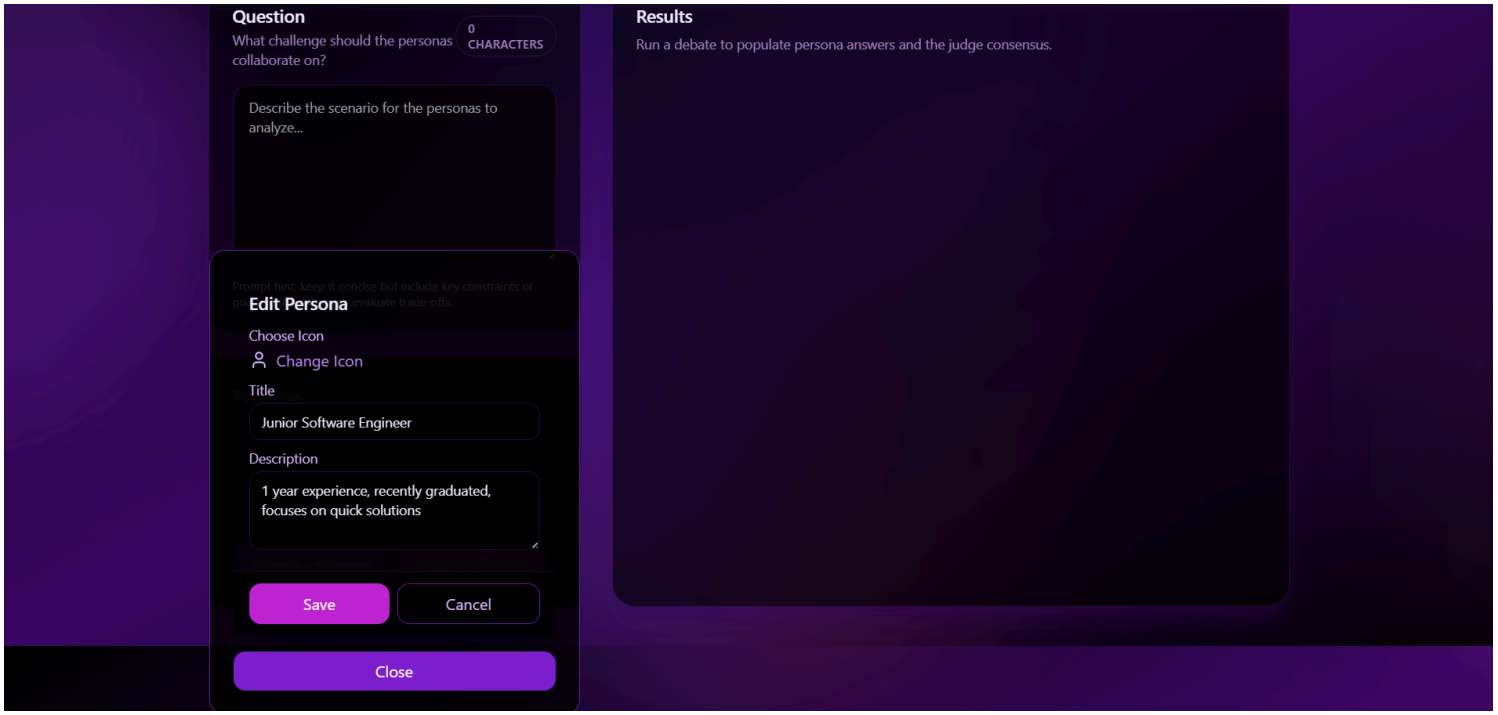

Figure 6: Edit persona

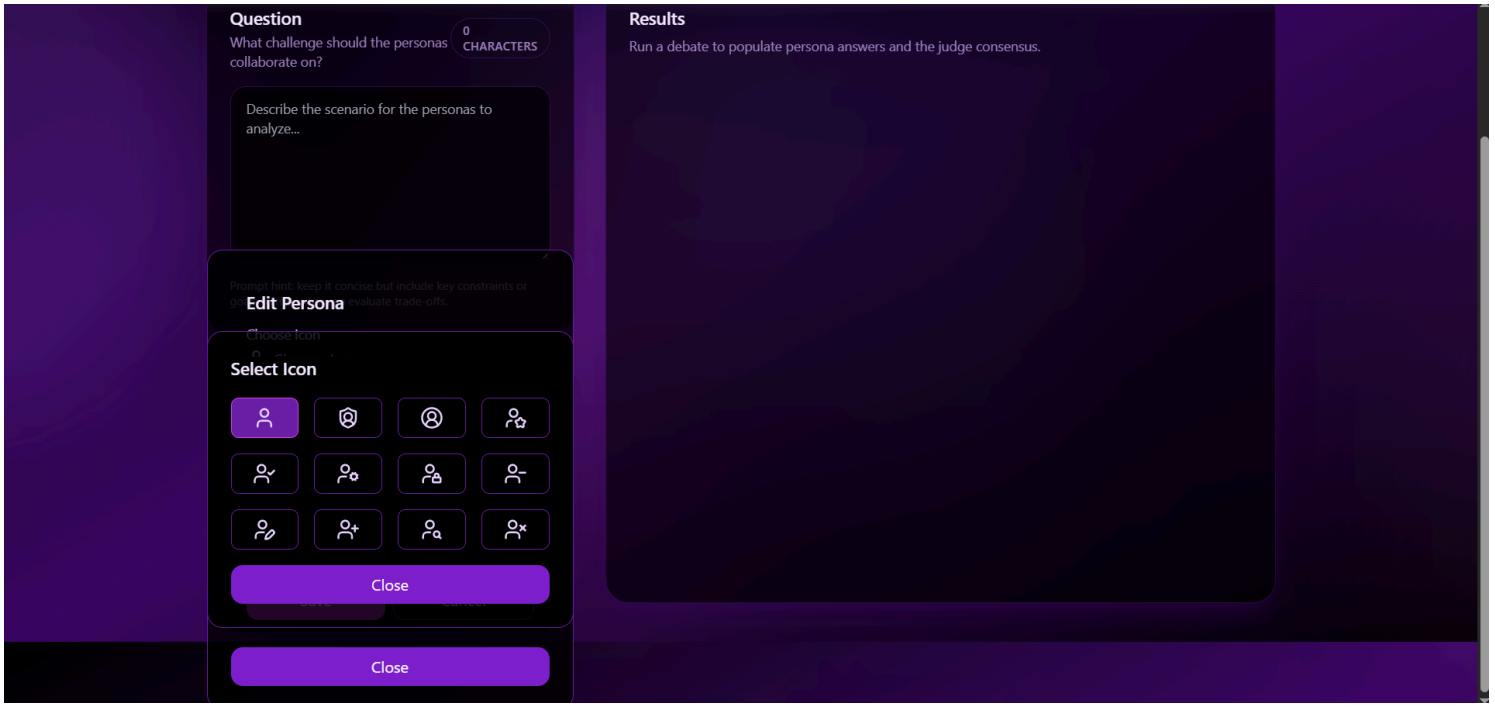

Figure 7: Select Persona Icon

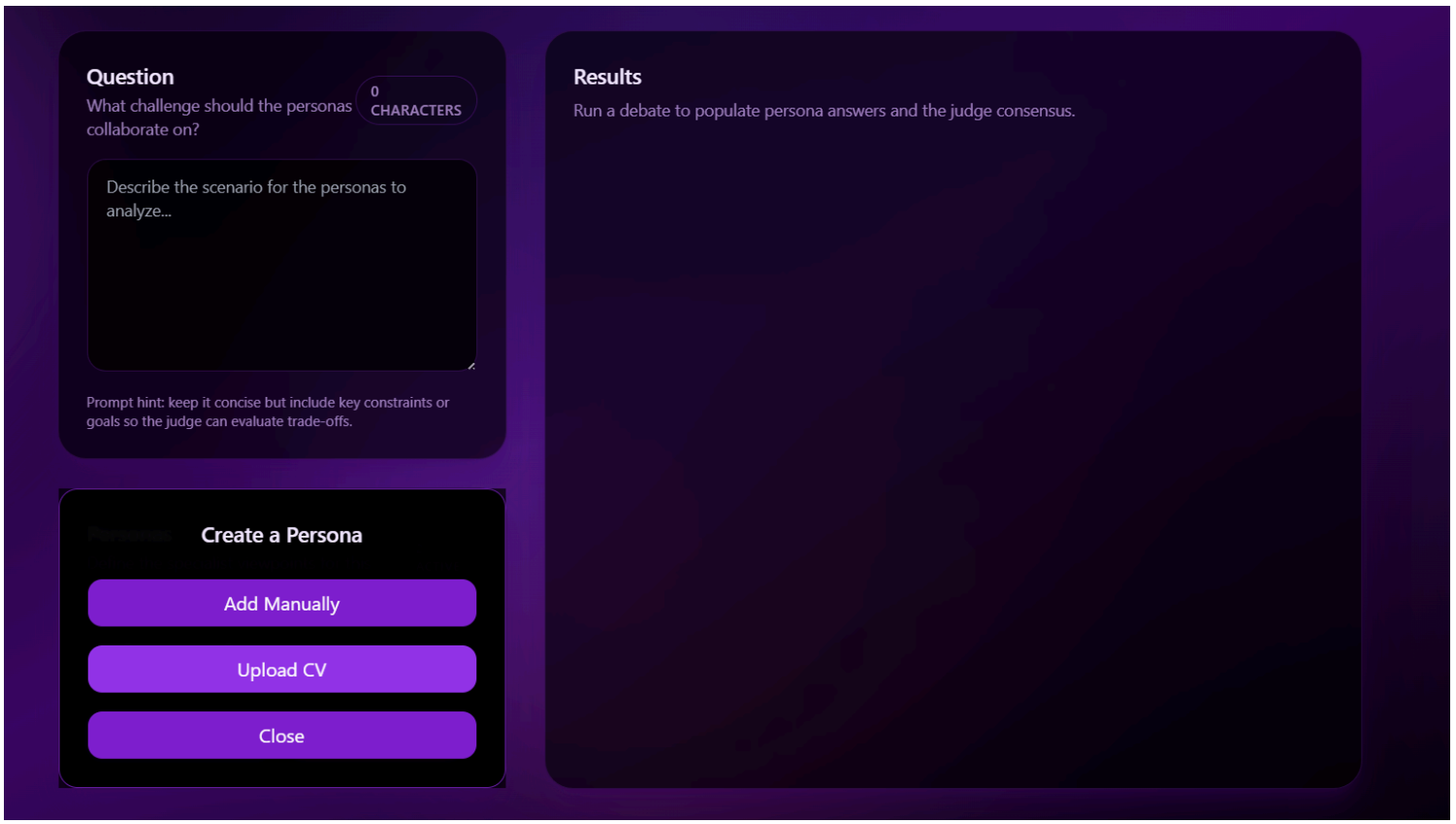

Figure 8: Create Persona

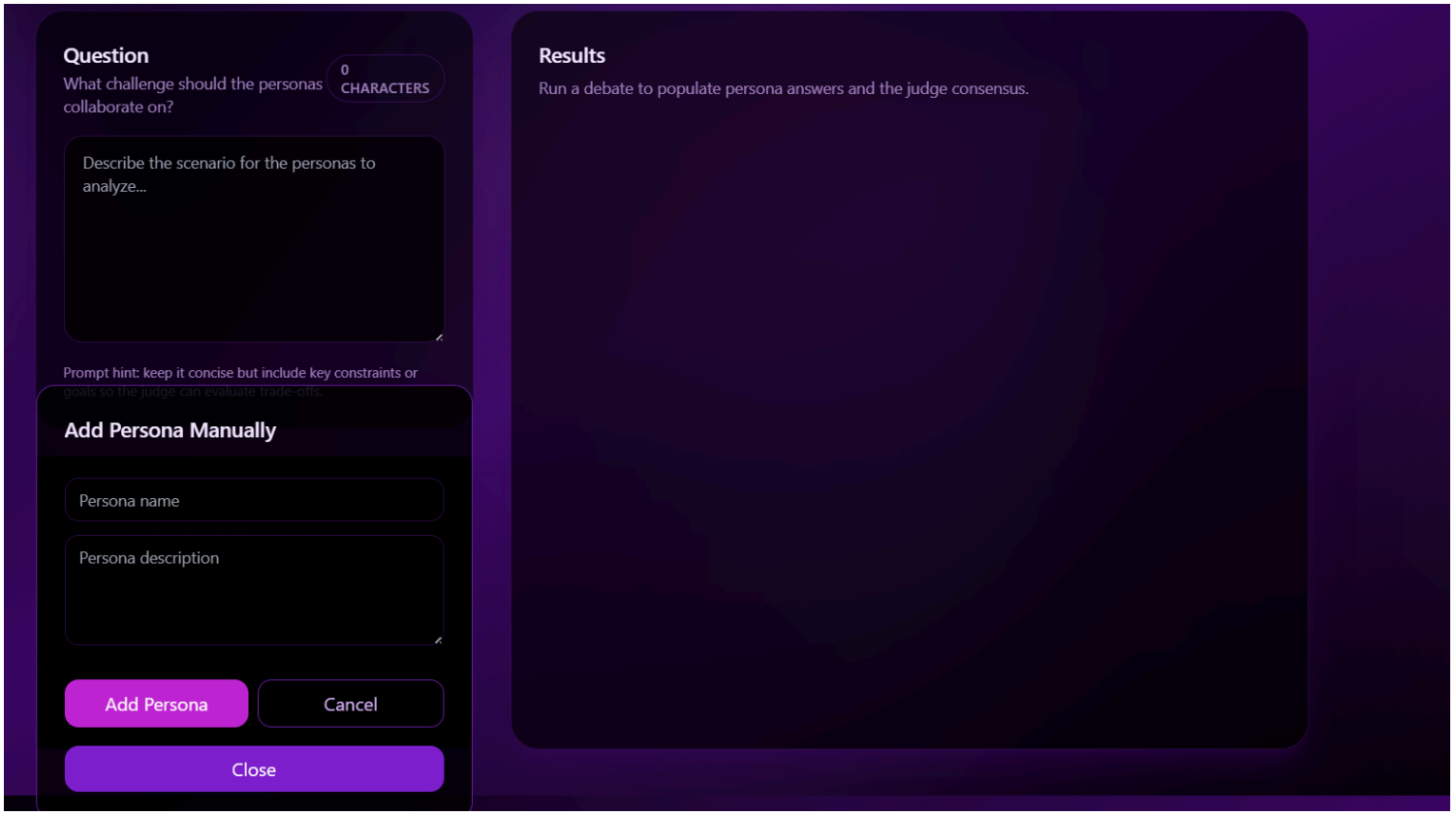

Figure 9: Create Persona Manually

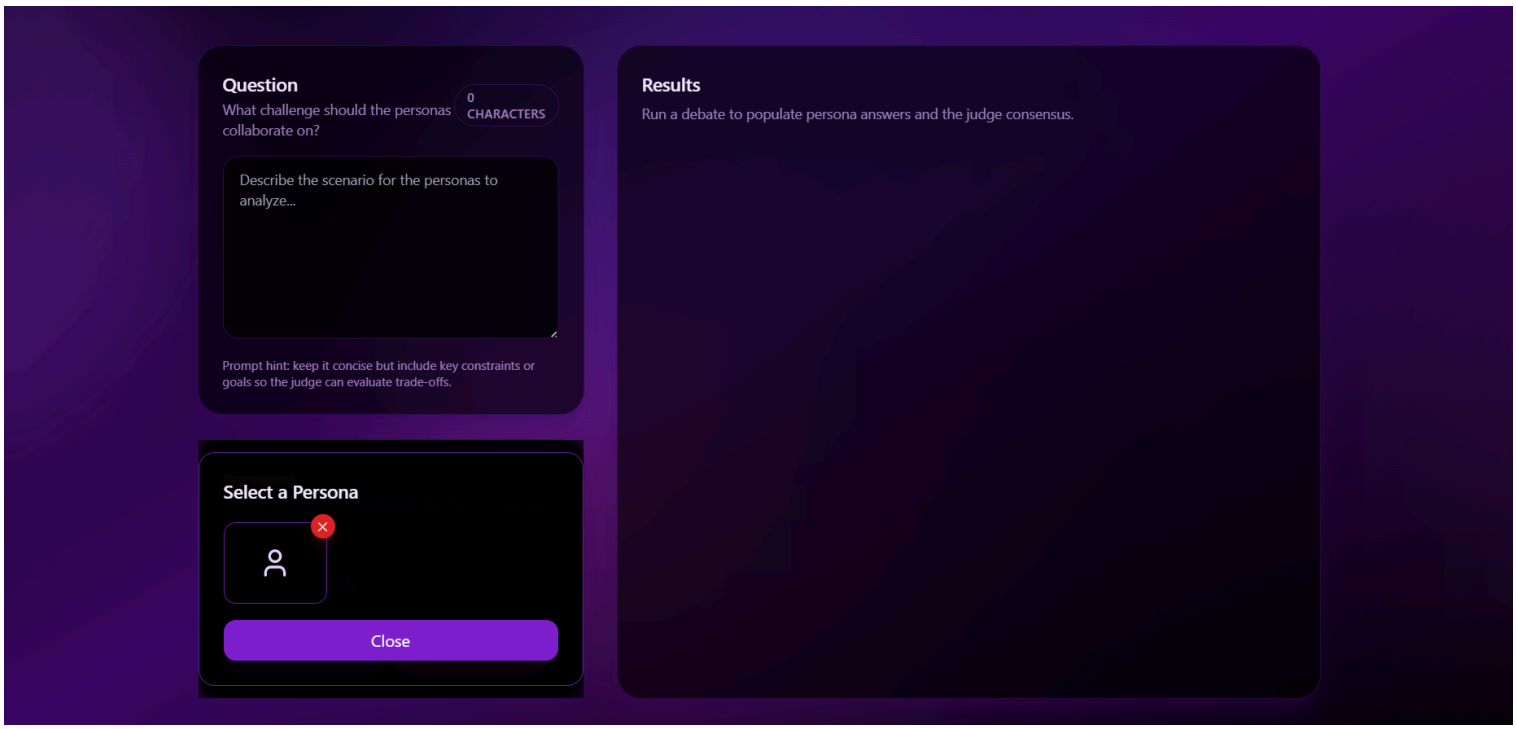

Figure 10: Select a Persona from created Personas to give its opinion with the other personas or delete it from the DB

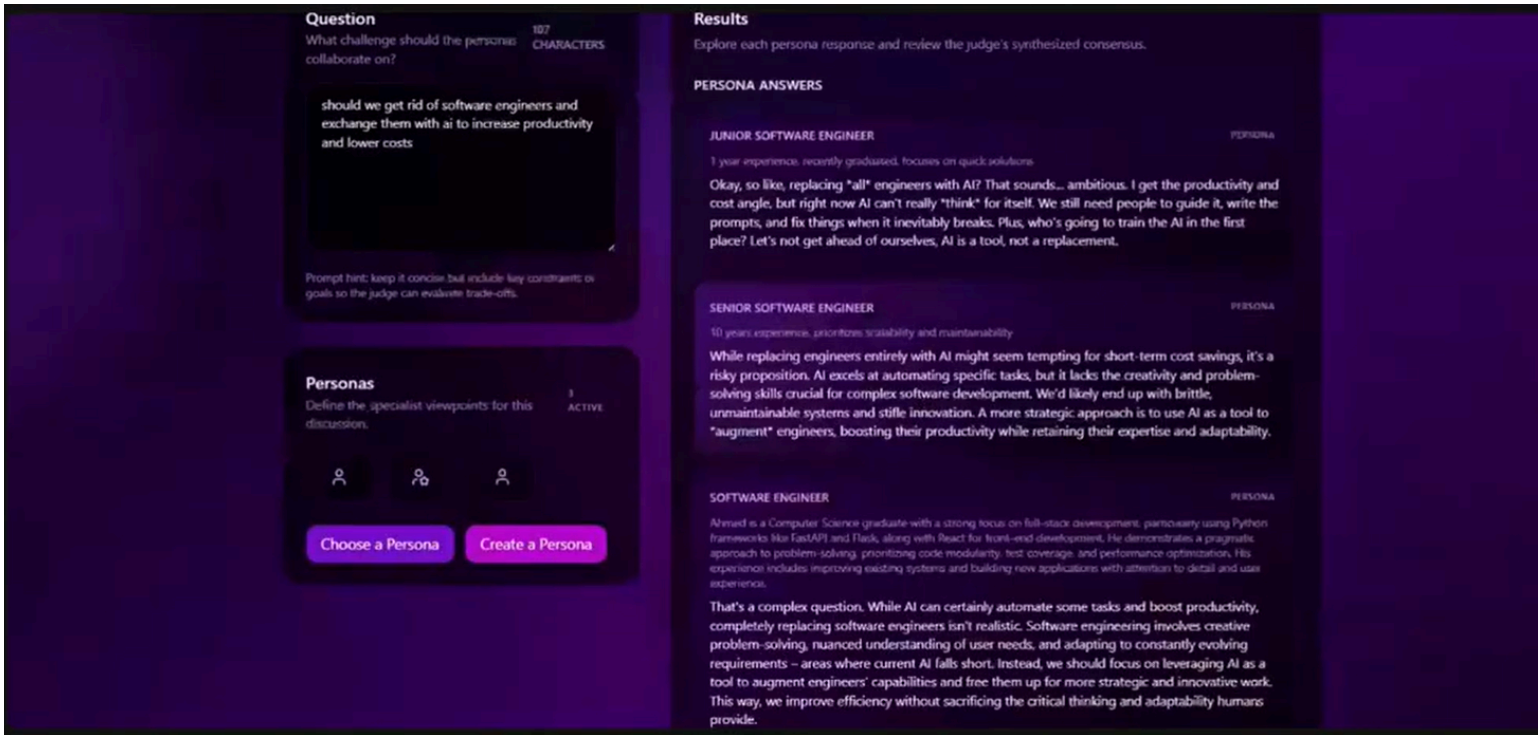

Figure 11: Question & feedback

4 Other Analysis Elements

4.1 Consideration of Various Factors in Engineering Design

We considered two major factors: project constraints and standards.

4.1.1 Constraints

4.1.1.1 Implementation Constraints

The implementation of Consensia is influenced by several technical limitations stemming from architectural decisions, technology stack, and available computational resources. These constraints define what is realistically achievable in the development timeline and influence both system performance and design choices.

Dependency on External LLM Providers

The system relies heavily on external APIs such as Google Gemini [3], OpenAI [2], and potentially other third-party AI services. Because these models run fully on cloud endpoints rather than locally:

1. Stable internet connectivity is required for all debate and persona generation processes.
2. Backend processing speed is limited by API response times, rate limits, and quota.
3. Any change in API availability, pricing, model versions, or request/response formats may require code refactoring.

Asynchronous Multi-Agent Execution Complexity

The debate pipeline requires:

- Multiple persona prompts,
- Sequential message passing,
- Judge evaluation,
- Asynchronous LLM calls.

similar to multi-agent LLM workflows studied in recent research [4], [5].

This introduces constraints such as:

- Increased complexity in request orchestration,
- Requirement for async-safe code on backend (FastAPI + asyncio),
- Risk of race conditions or partial failures if one persona fails to respond.

Frontend–Backend Contract

The system requires strong consistency between:

1. Persona schemas,
2. Task schemas,
3. Consensus request formats.

Because the backend validates all input using Python Pydantic schemas [7] :

- Any minor schema modification requires synchronized updates on the frontend.
- Mismatches between versions can cause request rejections or failed debates.

Limited Local Storage and Persona Management

Persona creation from:

- Prompts,
- CVs,
- Scraped data

requires storage and indexing.

Constraints:

- Storing raw documents (CVs, scrape data) increases disk requirements.
- Persona regeneration from large scraped datasets may be slow.
- Database must maintain consistency between personas, tasks, and debate logs.

Scraping & Data Processing Constraints

The scraping API introduces [4]:

1. Restrictions from website anti-bot measures,
2. Rate limitations and CAPTCHAs,
3. Ethical rules preventing scraping of restricted/private content,
4. Inconsistent formatting in external sources requiring preprocessing.

This limits the speed and reliability of data-driven persona creation.

Local Hardware & Development Environment

Since multi-agent interactions can require multiple LLM calls per debate, development environments face limitations such as:

1. Increased latency when running many personas,
2. Long-running async tasks during testing,
3. High costs and time delays when debugging LLM behavior.

Developers often need:

1. Stable network,
2. Sufficient CPU/RAM for running backend + frontend + development tools,
3. Keep-alive servers to prevent timeouts during multi-round debates.

Security & Privacy Safeguards

Handling user-uploaded CVs and documents requires:

1. Sanitizing all inputs,
2. Preventing raw file execution,
3. Ensuring no harmful content is passed between personas and judge.

These constraints limit:

1. Full freedom of persona creation,
2. File formats supported (e.g., PDF, DOCX but not executable files),
3. Ability to store long-term raw personal data.

Limited Team Size and Development Timeline

As a student project with a fixed timeline:

1. Implementation must favor modular, simple, maintainable components,
2. Certain advanced features (distributed workers, cloud autoscaling, heavy scraping) must be simplified,
3. Optimization is limited to what is feasible within academic deadlines.

4.1.1.2 Economic Constraints

Limited Budget

The project is conducted by a team of college students, limiting the budget of the project. The budget allocation for the project will be carefully planned, tracking and minimizing costs as necessary. We will also seek external funding from sources like ‘Microsoft for Startups’ [8].

Hardware Overhead

The application has a computationally intense process when running multiple LLMs which are all roleplaying and generating refined responses. As such, the user may be required to have strong hardware to run it. This process will be optimized to reduce computational overhead, to have a program suited to run in less advanced environments.

Cost of API Integrations

The application regularly executes LLM API calls such as the Gemini API [3], and may use other APIs like SerpAPI (Google Search API), significantly increasing operating cost of the project [4].

Licensing

The project will utilize various already existing technical tools like labeled datasets, which can often come with commercial licenses. We will carefully consider the licensing costs and choose which services are worth investing in, while utilizing open-source and academically licensed works as much as possible.

4.1.1.3 Ethical Constraints

Data Privacy and Consent

Sources such as CV's, research papers, or scraped contents to build a persona might violate privacy expectations unless the data is publicly available or reached by consent of a real person. Even publicly available data (LinkedIn, GitHub, academic websites) requires ethical considerations if the individuals did not explicitly consent to being modeled by AI. The system must ensure GDPR-style principles such as purpose limitations and storage security [9].

Bias Possibility and Fair Representation

Created personas might misrepresent the intended individual unless the description process is thoroughly shaped, as persona-based behavior can vary significantly between models [10]. Seniority-based stereotypes might create a biased environment (overestimating a professor's rigidity or underestimating a junior CS student's knowledge). Since the created personas debate and come with a consensus, biased personas might negatively affect the trust-worthy result.

Misinterpretation of Consensus

Users must be informed that; reached consensus is not an authoritative technical truth, disclaiming:

- Consensus is not guaranteed to be correct.
- Personas are simulated creations, not real experts in the field.
- The product should not be taken as the only authority for critical software decisions in sensitive fields.

Accountability and Transparency

The system must ensure accountability and transparency to users by providing what data is used to create a persona, how personas debate and the insights of created personas initial thoughts on a given question, and the consensus reasoning.

Ethical Persona Simulation

During persona creation process possible harmful or offensive inputs that result with violative outputs must be avoided. For simulating real individuals, avoiding impersonation must be ensured.

Table 1: Factors that can affect analysis and design.

	Effect level	Effect
Public safety	4	When the system is used for strategic software decisions, problematic consensus could lead to system failures.
Public health	0	No effects
Public welfare	0	No effects
Global factors	3	Language support should be considered in the case that there are foreign words in the labeled data.
Cultural factors	0	No effects
Social factors	7	LLMs often have inherent biases. It should be prompted to identify and mitigate biases in persona responses to ensure fairness.
Environmental factors	3	Discussion of the carbon footprint of high-compute LLM calls. It could be encouraged to use smaller, localized models for simpler tasks.
Economic factors	7	High token costs due to multiple persona generations and Judge LLM processing.

4.1.2 Standards

The project must comply with ethical and technical standards that ensure safe handling of personal data, transparent model behavior, and responsible use of real individuals' information when generating personas. All CV-based personas must be created with explicit consent, and any sensitive data must be minimized, anonymized, or excluded where possible [9]. The system should avoid reinforcing biases, ensure fairness across personas, and prevent harmful or misleading outputs during multi-agent debates. It should also maintain clear boundaries between real individuals and their simulated personas, ensuring that generated behaviors are probabilistic approximations rather than definitive representations. Finally, the platform must follow general software engineering best practices such as reliability, auditability, data security, and version-controlled experimentation to support trustworthy consensus generation.

4.2 Risks and Alternatives

4.2.1 Model Hallucination

The Judge reaches a consensus based on false information provided by a persona.

Alternative: No current alternative.

4.2.2 API Downtime/Rate Limiting

The system becomes unresponsive during persona generation.

Alternative: The System is executable with the pre-created personas and can still process when limiting situations appear.

4.2.3 Majority Bias

The Judge simply picks what most persona say, even if the minority is the only one who is correct.

Alternative: No current alternative.

4.3 Project Plan

Table 2: List of work packages

WP#	Work package title	Leader	Members involved
WP1	CS491 Meetings and Reports	-	All members

WP 1: CS491 Meetings and Reports			
Start date: *17.10.2025* **End date:** *19.12.2025*			
Leader:	*No elected leader.*	**Members involved:**	*Ahmed Haikal* *Hakan Karakoç* *Amirhossein Ahani* *Türker Köken* *Mehmet Hakan Yavuz*
Objectives: *Determining a project topic that is innovative and feasible. Receiving feedback from the course instructors, supervisor and innovation expert. Preparing all the required reports for the CS491 course.*			
Tasks: ***Task 1.1 Selecting a project topic and supervisor*** **:** *We discussed our ideas with numerous professors and selected a topic and a supervisor that we believed to be the most innovative and is suited for the computer engineering fields we are interested in.* ***Task 1.2 Discuss the selected project and its specifications*****:** *We had meetings as a group to fill out the Project Information Form and the Project Specifications Document.* ***Task 1.3 Prepare a presentation for the Innovation Expert*****:** *We had a meeting with an innovation expert for them to assess our project in terms of innovativeness. We used the feedback we received to fill out the Innovation Expert Evaluation Form.* ***Task 1.4 Attend project progression meetings*****:** *We had meetings with the course instructors to keep them updated on our progression and receive feedback.* ***Task 1.5 Attend project seminars*****:** *We attended the seminars organized for the CS491 course, in order to learn various aspects of designing a project (Role of Documentation, Property Rights, AI, etc.). We reflected on our project with the new information we learned.* ***Task 1.6 Discuss the project with the supervisor regularly*****:** *We had meetings with our supervisor to keep them updated on our progression and receive feedback.* ***Task 1.7 Implementation*****:** *The project was implemented at an introductory level in order to demonstrate its core functionalities during the meetings.* ***Task 1.8 Finalize the Analysis and Requirements Report*****:** *The last deliverable is finalized in order to satisfy the course requirements.*			
Deliverables ***D1.1: Project Information Form*** ***D1.2: Project Specification Document*** ***D1.3: Innovation Expert Evaluation Form*** ***D1.4: Analysis and Requirements Report***			

4.4 Ensuring Proper Teamwork

- Each member of the group is expected to attend planned meetings.
- Each member of the group will be an active participant of the project development process.
- Each member of the group should research the assigned task or work on the part they desire, as long as it is discussed and favoured by the group.
- Group members should help each other when possible and maintain clear, consistent communication with the instructor.

4.5 Ethics and Professional Responsibilities

- **Transparency**: It should be stated that the responses are generated with AI-based personas to avoid deceiving users.
- **Data Privacy**: It should be ensured that uploaded CVs or scraped data are handled according to the privacy standards.
- **Accountability**: It should be stated that, “Judge” consensus is a recommendation tool, not a final decision-maker.

5 Glossary

API: Application Programming Interface

AI: Artificial Intelligence

LLM: Large Language Model

CV: Curriculum Vitae

GDPR: General Data Protection Regulations

CTO: Chief Technology Officer

QA: Quality Assurance

SRE: Site Reliability Engineering

6 References

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.

[2] OpenAI, “GPT-4 Technical Report,” 2024. [Online]. Available: https://openai.com

[3] Google DeepMind, “Gemini: A Family of Highly Capable Multimodal Models,” 2024.

[4] SerpAPI, “Google Search API Documentation,” 2024. [Online]. Available: https://serpapi.com

[5] A. Yang, Y. Bai, et al., “Large Language Model Agents,” arXiv preprint arXiv:2401.11016, 2024.

[6] C. Chen, S. Zhao, et al., “Multi-Agent Debate Improves Reasoning in LLMs,” arXiv preprint arXiv:2305.14387, 2023.

[7] Pydantic Developers, “Pydantic v2 Documentation,” 2024. [Online]. Available: https://docs.pydantic.dev

[8] Microsoft, “Microsoft for Startups Founders Hub,” 2024. [Online]. Available: https://www.microsoft.com/startups

[9] European Union, “General Data Protection Regulation (GDPR),” Official Journal of the European Union, 2016.

[10] R. Xu, X. Li, et al., “Persona-Based Evaluation of LLM Behavior,” arXiv preprint arXiv:2403.01245, 2024.

[11] GitHub, “GitHub Copilot Technical Documentation,” 2024. [Online]. Available: https://docs.github.com/en/copilot