University of Wollongong

IRRATIONAL GENERATIVE AGENTS

A Report submitted as a part of
the Assignment for the subject CSIT998

Capstone Project

from

UNIVERSITY OF WOLLONGONG

by

Yifei Wang - 8339715
Karan Goel - 7836685
Koyuki Abe - 8031046
Chang Liu - 8108742
Nowmin Naj Manisha - 8248862
Monyridh Yady - 8358898

School of Computing and Information Technology
Faculty of Engineering and Information Sciences

2025

IRRATIONAL GENERATIVE AGENTS

Capstone Project

School of Computing and Information Technology
University of Wollongong

ABSTRACT

Irrational Generative Agent is a cognitive agent framework designed to simulate both rational and irrational facets of human thinking. Grounded in dual-process theories from cognitive science, the architecture integrates System 1 (fast, heuristic reasoning) and System 2 (slow, deliberative reasoning), enriched with internal traits such as personality, emotion, memory, and cognitive biases.

Agents exhibit dynamic, evolving behavior that mirrors human cognitive tendencies. A lightweight, interactive simulation environment supports scenario customization and real-time observation. The framework is evaluated through behavioral human-likeness assessments, longitudinal cognitive development tracking, and analysis of bias-influenced decision-making. This project provides a reusable foundation for building and studying cognitively plausible artificial agents.

Keywords: Human-AI interaction, generative agents, large language models, cognitive architecture, dual-process theory, simulation environment

PROJECT RESOURCES

- **Irrational Agent simulation:** `https://shorturl.at/tvEyF`
- **Source Code:** `https://github.com/orgs/Irrational-Agents/repositories`

Video Demo : https://shorturl.at/x7X7o

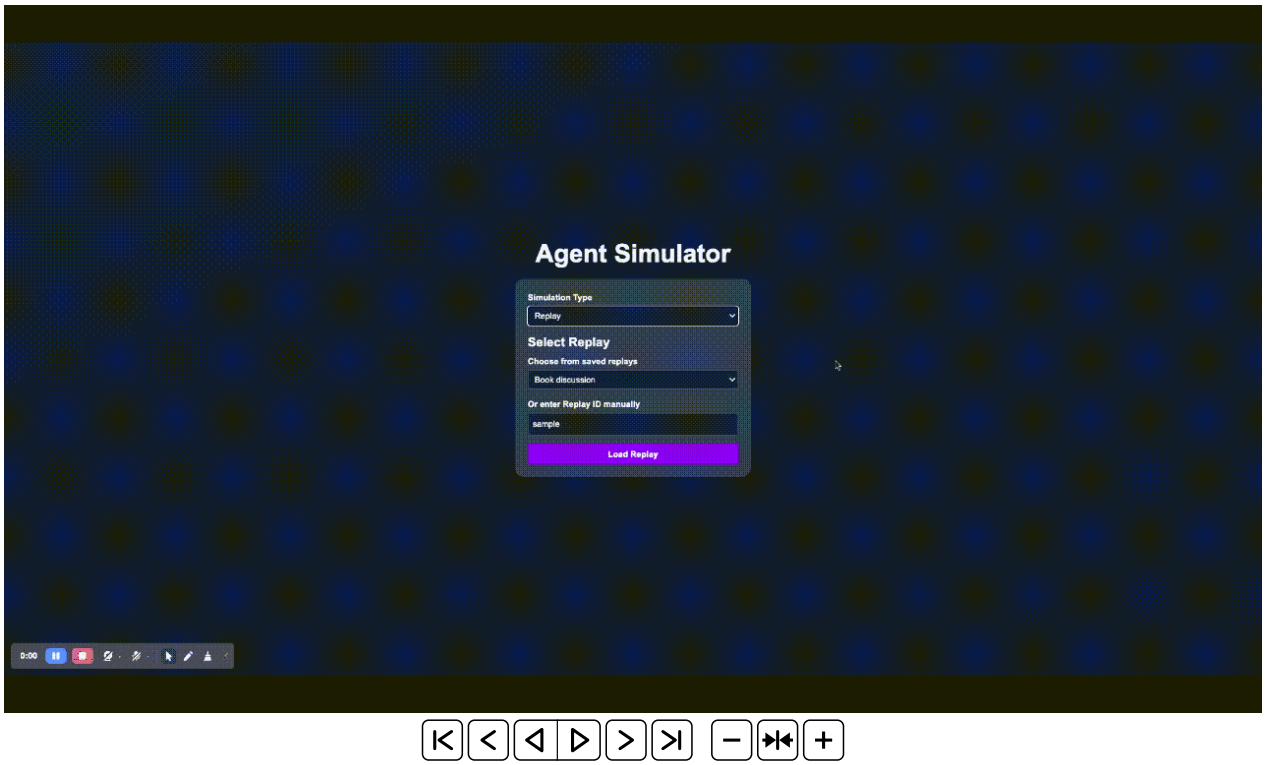
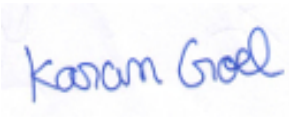
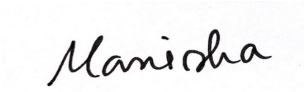
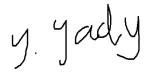


Figure 1: Full Demo (works with Acrobat)

MEMBER **CONTRIBUTION**

Table 1: Contribution Table

Name	Student Number	Contribution	Signature
Yifei Wang	8339715	Contributed	
Karan Goel	7836685	Contributed	
Koyuki Abe	8031046	Contributed	
Nowmin Naj Manisha	8248862	Contributed	
Chang Liu	8108742	Contributed	
Monyridh Yady	8358898	Contributed	

Contribution Summary:

- **Yifei Wang (Backend Developer):** Led system architecture, implemented core Agent modules, handled system integration, evaluation, authored technical documentation and ensured overall documentation quality control.
- **Karan Goel (Full-stack Developer):** Developed frontend and backend features, handled system integration, deployment pipeline, evaluation, authored system implementation documentation.
- **Koyuki Abe (Project Manager / Research Developer):** Research proposal, initial planning, system architecture and backend design, setup and implemented prototype agent flow and documented methodology.
- **Nowmin Naj Manisha (Assistant Project Manager / Project Manager):** Executed comprehensive project management duties that included requirement analysis, schedule management, authored management documentation and support research and implementation.
- **Chang Liu (Documentation Coordinator):** Participated in literature review, contributed to proposal, report writing, formatting, and UI scene sketching.
- **Monyridh Yady (Documentation Assistant):** Contributed to literature review, proposal development, report writing.

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1 Background

Modeling human behavior has long been a major challenge in disciplines such as psychology and artificial intelligence [1]. Although traditional methods—like surveys, longitudinal studies, and lab-based experiments—offer important insights, they are often expensive, slow to yield results, and face ethical hurdles [2]. Observing phenomena like personality development, emotional regulation, and cognitive bias typically involves complex variables and can require decades of data [3, 4].

At the same time, rapid advances have occurred in the development of virtual agents used in games, educational software, and simulation platforms [5]. However, the majority of these agents are confined to executing pre-programmed behaviors, lacking the deeper psychological constructs—like memory, bias, or adaptive personality—that characterize real human cognition [6]. This gap highlights the need for more sophisticated models capable of reflecting not just decisions, but the evolving, sometimes irrational reasoning that leads to them. A system that is between rational and irrational that can simulate human behavior.

1.2 Overview

This project proposes a novel approach to studying human-like decision-making by replicating internal cognitive mechanisms in simulated agents. Unlike traditional research reliant on human or animal subjects—which is costly and ethically sensitive—our framework allows scalable testing using autonomous agents. Drawing from Kahneman and Tversky's dual-process theory, the system distinguishes between intuitive and deliberate reasoning processes, capturing how biases and context affect behavior [7].

The architecture combines a Python-based decision model with a Unity-powered 2D environment

and natural language generation from GPT-4o, enabling researchers to craft scenarios and monitor agent reactions under controlled conditions. By simulating responses to environmental stressors and social interactions, this setup enables low-cost, high-fidelity experimentation in behavioral modeling [8]. It opens new doors for the study of irrationality, personality drift, and emotionally driven behavior in agent systems.

1.3 Contributions

This work presents a cognitively inspired simulation framework for modeling human-like behavior in virtual agents. All system components—from architecture to memory and planning logic—were custom-built to enable realistic, adaptive behavior.

The main contributions are as follows:

- We develop an **irrational agent architecture** that merges sensory input, motivation, emotion, personality traits, and cognitive biases. Central to this architecture is a **dual-process model** [9], which separates fast, intuitive responses (System 1) from slower, reflective reasoning (System 2), enabling agents to display impulsive, contradictory, or suboptimal behavior.
- We propose a **modular framework** where cognitive bias modules—such as loss aversion, time discounting, and confirmation bias—can be selectively activated to study their effect on decision-making in controlled settings.
- We implement a **lifelike agent loop**, where entities perceive, evaluate, and act based on internal states. Over time, agents update their memory and gradually shift personality traits, allowing for dynamic growth and long-term behavioral evolution.
- We construct a persistent, visually grounded simulation world (see Figure 1.1) using Unity. This environment enables **real-time observation** of agents' decisions, emotional fluctuations, and interactions, all accessible via a web-based interface.

Figure 1.1: Simulation World

2. Project Management

2.1 Product Scope

The *Irrational Agents* system simulates human-like behavior through autonomous agents powered by GPT-4 Mini and integrated cognitive modules. The product scope is defined through three perspectives:

2.1.1 Included Features

The current implementation includes:

- **Environment**: Static 2D town visualization (Phaser/WebGL) with observation-only interface
- **Agent Architecture**:
  - Personality, emotion, and memory modules influencing decision-making
  - Planning system generating daily schedules via GPT-4 Mini
- **Social Dynamics**: Autonomous interactions (greetings, conversations, avoidance)
- **System Infrastructure**:
  - Modular backend for reasoning components
  - Behavior logging in CSV/JSON formats

2.1.2 Excluded Features

The following are deliberately excluded from v1.0:

- **Environment**: Dynamic maps, scene transitions, or user customization

- **Memory**: Persistent storage across sessions
- **Interaction**: Real-time user control or multi-user features
- **Model Flexibility**: Alternative LLM integrations

2.1.3 Future Road map

Planned enhancements include:

- **Extended Cognition**: Long-term memory persistence
- **Dynamic Content**: Event-driven scripting and custom maps
- **Technical Expansion**:
  - Multi-LLM support via plugins
  - Performance scaling (50+ agents)
- **Research Tools**: Customization UI and collaboration features

2.1.4 Work Breakdown Structure

The Work Breakdown Structure (WBS) for the Irrational Agents project divides the development process into five main phases: initialization, system design, agent and experiment development, evaluation and reporting, and presentation. Each phase is subdivided into manageable tasks to ensure clarity and systematic progression. Figure 2.1 represents a visual of the structure. This WBS ensures that each development milestone contributes directly to the refinement of a robust and behaviorally grounded irrational agent.

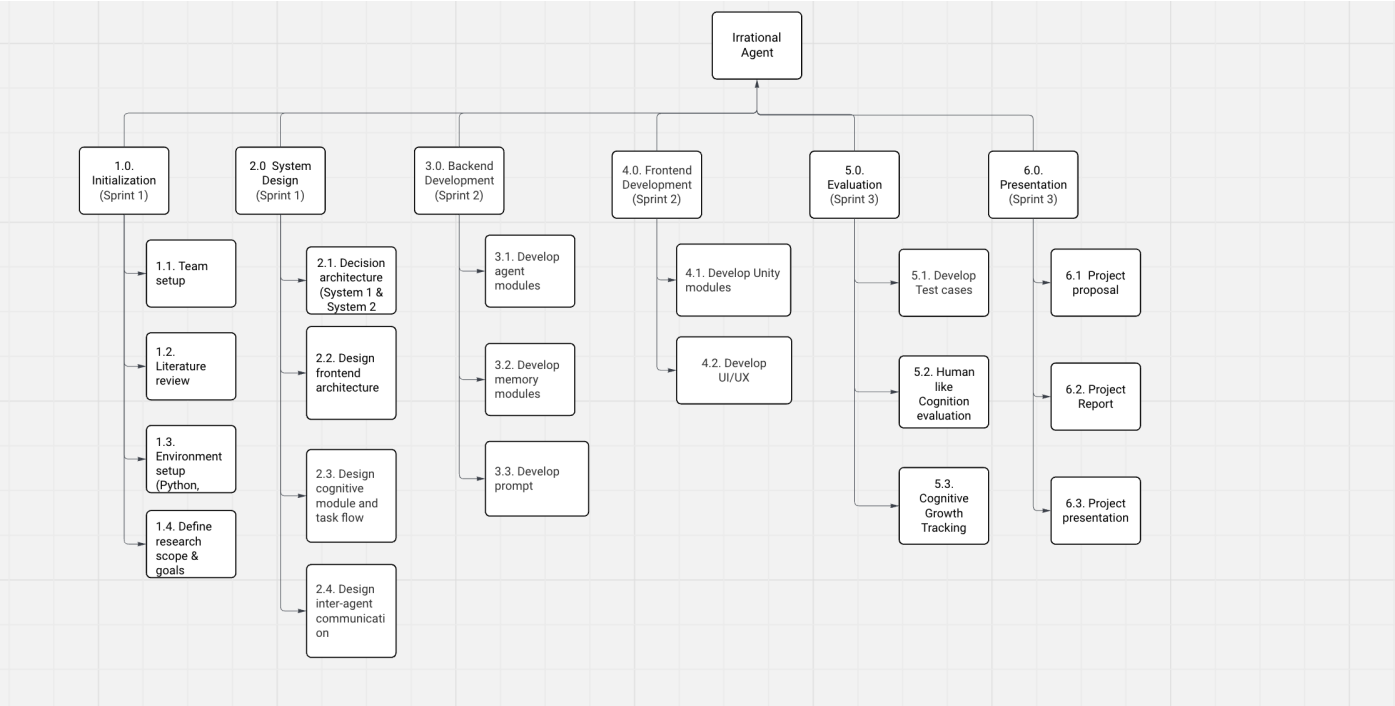


Figure 2.1: WBS for Irrational Agent

2.2 Requirements Specification

The Irrational Agent was guided by a set of general, functional, and non-functional requirements. Functionally, the system needed to model both System 1 (intuitive) and System 2 (analytical) thinking, simulate psychological biases like loss aversion and framing effects, and allow agents to learn and adapt through memory and feedback loops. It also needed to support flexible experiment setups, live visualization, and detailed logging for analysis.

Non-functional requirements included real-time performance to manage multiple agents without lag and reproducibility to ensure consistent results under identical conditions. Secure data handling and a user friendly sandbox environment were also important to maintain experiment integrity. Together, these requirements shaped the development of a system capable of simulating human like irrational decision making in a controlled environment.

2.2.1 General Requirements

Virtual Experiment Environment Construction

- Development of AI agents capable of reproducing various psychological states, personalities, and symptoms of mental disorders.
- Functionality for quantifying (parameterizing) responses to external stressors and therapeutic interventions.
- Breakdown and modularization of decision-making processes.

Experiment Design Interface

- User-friendly interface allowing researchers to easily set experimental parameters and interact with NPCs.
- Features and extensibility to support complex experimental designs and long-term research.

2.2.2 Functional Requirements

Table 2.1: Functional Requirements

ID	**Requirement**	**Priority**	**Mapped WBS tasks**
FR-ENV-01	Render 2D town environment via Phaser WebGL	High	4.0
FR-AGT-01	Configure agent personality/emotion parameters pre-simulation	Medium	3.0
FR-AGT-02	Generate daily schedules using LLM	High	3.0
FR-SYS-01	Update agent reasoning on simulation tick	Critical	3.0
FR-UI-01	Provide read-only web observation interface	High	4.0

FR-SOC-01	Agent to agent social interactions	High	3.0, 4.0
FR-MEM-01	Maintain session-scoped memory affecting behavior and decision	Medium	3.0, 4.0
FR-ANA-01	Export structured logs in structured format (CSV/JSON)	Low	3.0, 4.0
FR-BIAS-01	Modular injection of biases in agent logic	High	3.0
FR-DUAL-01	Dual process reasoning system (System 1 and 2)	High	3.0, 4.0
FR-PROMPT-01	Interface with LLM for thoughts/plans/interactions	High	3.0, 4.0
FR-MULTI-01	Support for multi-agent communication and collaboration	Medium	3.0, 4.0
FR-VISUAL-01	Visualize agent state (mood, memory, current plan)	Hight	4.0
FR-CTRL-01	UI for controlling simulation and triggering experiments	High	3.0, 4.0
FR-DEPLOY-01	Support deployment via GitHub CI/CD pipelines	High	2.0
FR-LOGIC-01	Agent plans influenced by memory. Emotion, environment	Hight	2.0, 3.0, 4.0
FR-FEEDBACK-01	Agent traits evolve over time (feedback loop, personality drift)	High	3.0, 4.0, 5.0
FR-UI-02	Show interaction heatmaps and UI display enhancements	High	4.0, 5.0
FR-PRES-01	Generate figures and diagrams for presentation	High	6.0

2.2.3 Non-Functional Requirements

Table 2.2: Non-Functional Requirements

ID	Requirement	Category
NFR-PER-01	Support 20+ concurrent agents in browser	Performance
NFR-ARC-01	Modular architecture for LLM expansion	Extensibility
NFR-SEC-01	Local data processing only	Security
NFR-USR-01	Browser-based, no-install execution	Accessibility
NFR-MNT-01	Documented, modular codebase	Maintainability

2.3 Use Case Analysis

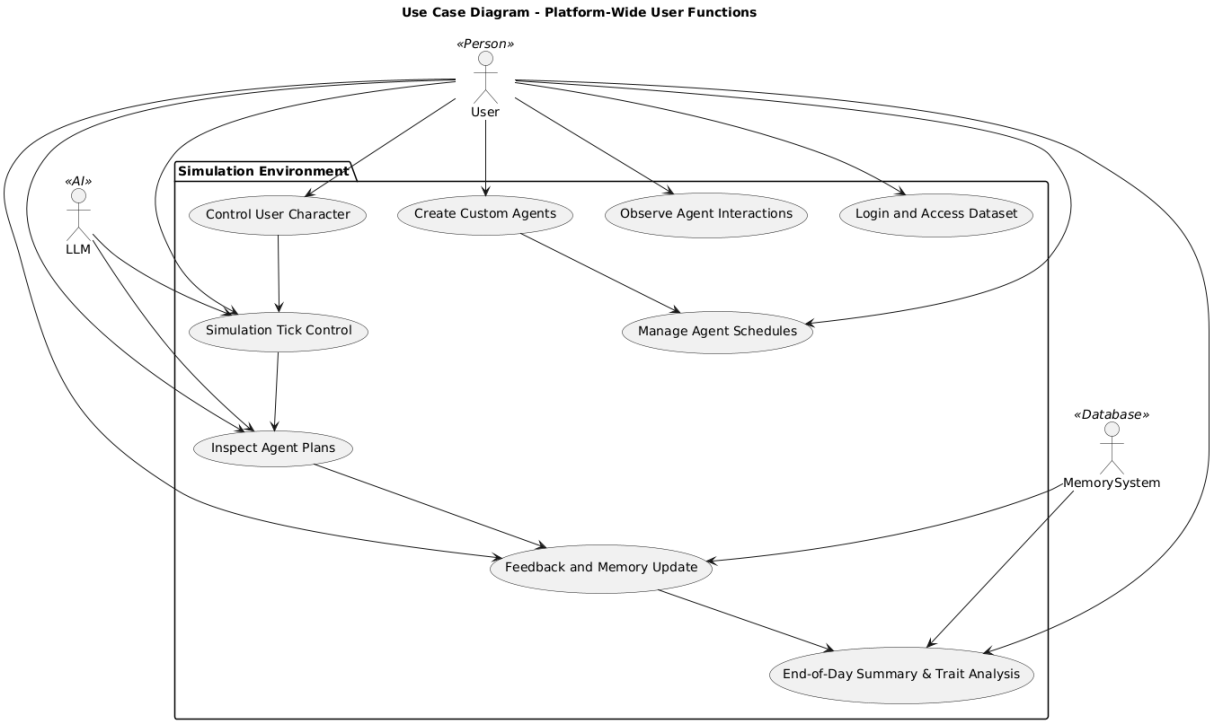


Figure 2.2: Core Simulation Use Cases

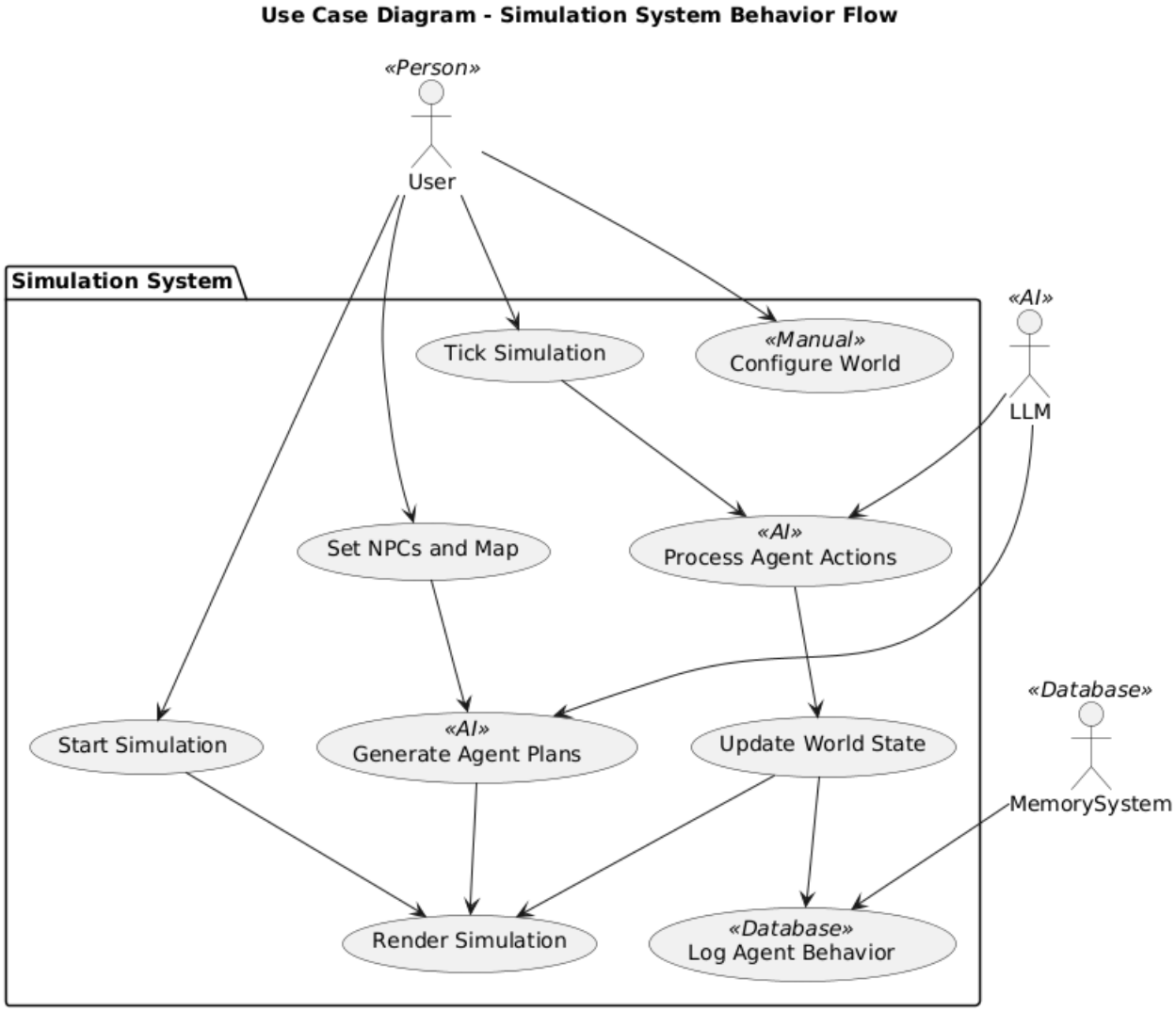


Figure 2.3: User Interaction Workflow

Key Use Cases:

- UC-01: Simulation Initialization
- UC-02: Autonomous Agent Behavior Cycle
- UC-03: Observation Data Export
- UC-04: Emotional State Update

2.4 Teamwork

2.4.1 Time Arrangements

This project follows an Agile Scrum development methodology that begins in August 2024 and ends on May 2025, spanning two academic semesters and a summer break. It is structured into unique phases. The first week is dedicated to research, literature review, and system design, while the second week focuses on development, integration, and deployment, and the last week focuses on testing and reporting. The first semester focuses mainly on the project proposal, application design and system development, with the time allocation detailed in Figure 2.4. The second semester emphasizes system deployment and report preparing. This structured section will provide a clear explanation of the second phase in Figure 2.5.

The application development cycle is designed to have three sprints:

1. Sprint 1: Exploration stage and design stage (August - October 2024)

   The first sprint starts on August 2024 and ends on October 2024, lasting 100 days with full team participation. This phase established the research foundation and overall architecture of the system.

   - Review of the literature on generative agents, dual-system theory, and behavioral psychology by Stan and Yadi.
   - Requirement analysis and experiment planning by Yuki and Manisha.
   - Planning and design architecture based on integration of biases, System 1 and System 2 modeling by Ephie, Karan, Manisha and Yuki.
   - Setup of development environments (Python, Phaser, Github repository) by Karan, Ephie ani Yuki.
   - Focused research into behavioral economics and human biases by Yadi and Manisha.
   - Weekly presentation on the research of literature review by Yadi and Stan.

- Develop project proposal by Stan and Yuki.
- Developed user stories based on selected psychological phenomena, linking academic theory to interactive behaviors in the simulation by Stan and Manisha.
- Design all the necessary architecture by Yuki, Manisha, Karan and Ephie.

2. Sprint 2: Coding Development stage (November 2024 - April 2025)

   The second sprint starts on November 2024 and ends on April 2025, lasting 150 days with Yuki, Karan, Ephie and Manisha's participation. This phase includes active development of the front end, personality, prompt and develop evaluation setup.

   - Developed the bias modules, agent module, prompt, front end and dual-process decision engine by Karan, Ephie, Manisha and Yuki during summer break.
   - Network architecture is build by Ephie and Karan during summer break.
   - Yuki and Ephie implemented agent personalities.
   - Yuki, Manisha Karan and Ephie designed and tested LLM prompt templates.
   - Karan and Yuki built and refined the Phaser sandbox UI and experiment scenes.
   - Karan and Ephie integrated WebSocket communication between Phaser and Python.
   - Karan, Ephie conducted building logging systems for the experiment.

3. Sprint 3: Validation, Testing and Reporting (April - May 2025)

   The third and final sprint starts on March 2025 and ends on May 2025, lasting 30 days with full team participation. Although Yadi, Stan and Manisha started gathering relevant documents, gantt chart, wbs, user story from November 2024. The final phase focused on conducting testing, evaluation and combining all the necessary documents in the report from previous semester and current semester.

   - Develop test cases and parallel document all the finding by Ephie and Karan.
   - Asses evaluation metrics and document all the findings by Yuki, Manisha, Kran, Yadi, Stan and Ephie.

- Addresses bug fixes and final system optimizations.
- Wrote and finalized the project report, covering system design, methodology, results, and evaluation with the help of all team members.
- Visual diagrams, charts, and experiment logs are created.

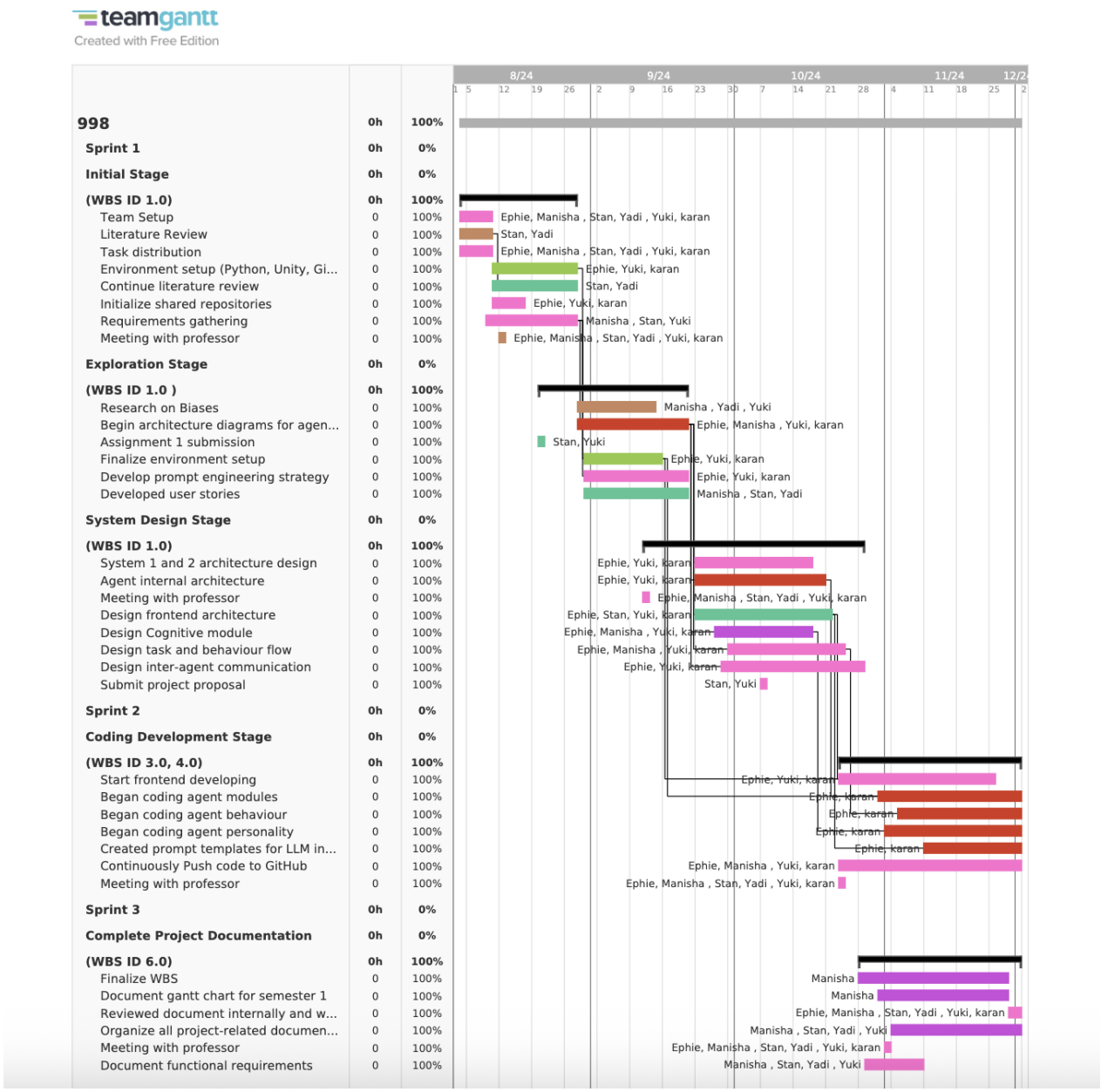


Figure 2.4: Semester 1 Gant chart

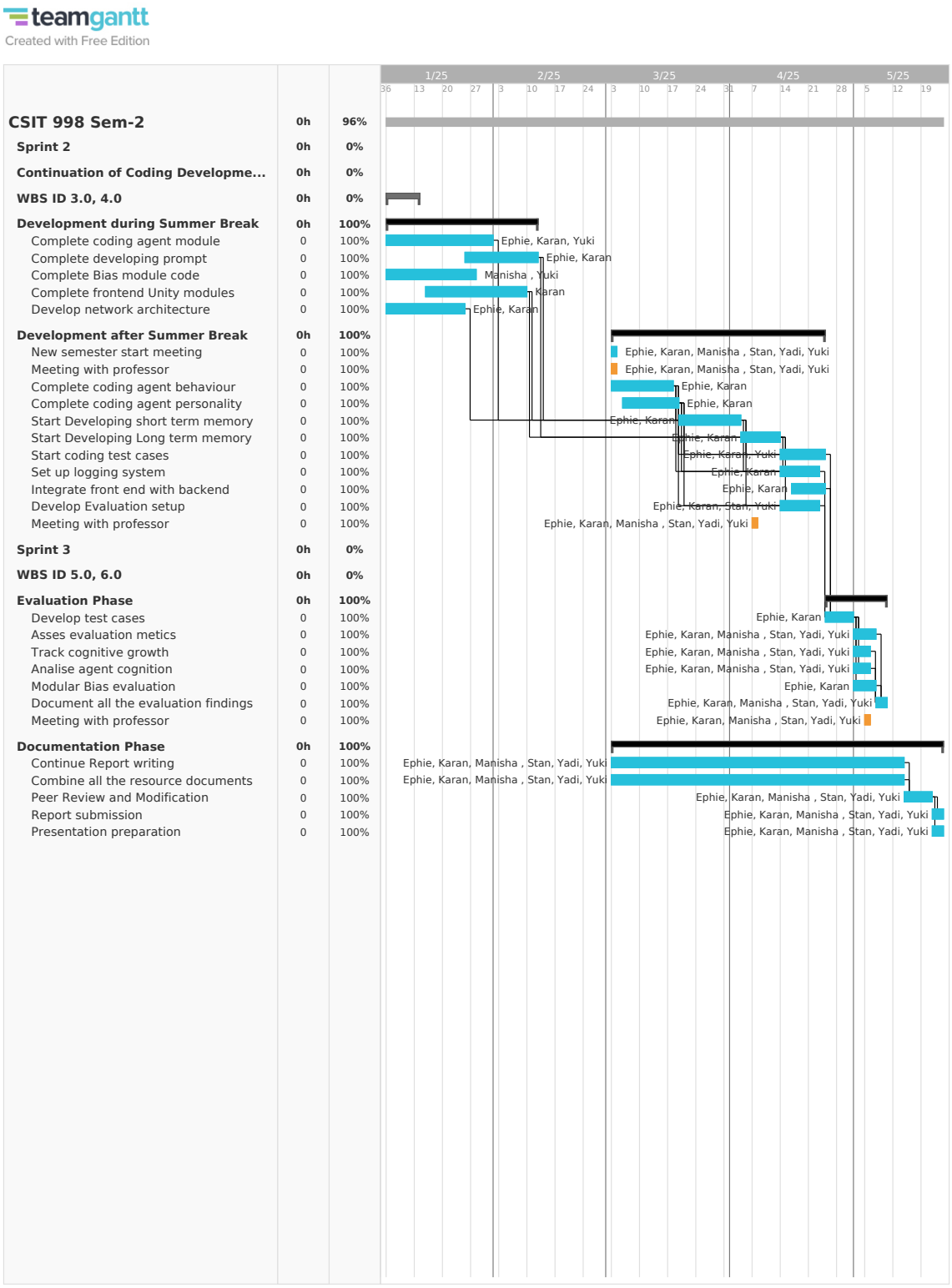


Figure 2.5: Semester 2 Gant chart

2.4.2 Project Monitoring

To ensure the Irrational Agent project adhered to its timeline, met objectives, and maintained high-quality standards, a structured monitoring approach was implemented. The monitoring strategy combined Agile-inspired methods with modern project management tools, tailored to the team's workflow and communication preferences. The following is an overview of the monitoring framework used throughout the project.

2.4.2.1 Development Method

The team adopted the Agile scrum approach that helps prioritize flexibility and iterative feedback.

- Sprint-Based Development: The project was structured into sprints every week. In each sprint, we focused on a specific set of features or tasks. This approach enabled development efforts and ensured that progress was regularly evaluated. The sprints were aligned with once in two week supervisor meetings, ensuring that the team received feedback and clear direction before and after each sprint.

- Weekly Team Meeting: Instead of daily stand-ups, the team conducted weekly on-sight meetings in university library. When a team member can not attend the face-to-face meeting , they can join with Google meet. These meetings were used to discuss progress, address challenges, and adjust tasks as needed. The weekly meeting allowed for greater flexibility while accommodating academic commitments.

- Biweekly Supervisor Meetings: Every two weeks, the team met in person with the project supervisor. These sessions were essential for getting feedback from the professor, which helped refining the project's direction and ensuring alignment with academic requirements.

2.4.2.2 Project Management Tools

To efficiently manage development tasks and team collaboration, the project leveraged a combination of tools suited to the team's workflow:

- GitHub Project Tracker: GitHub Figure 2.6 served as the primary tool for code management and issue tracking. It enabled structured version control and streamlined task management.

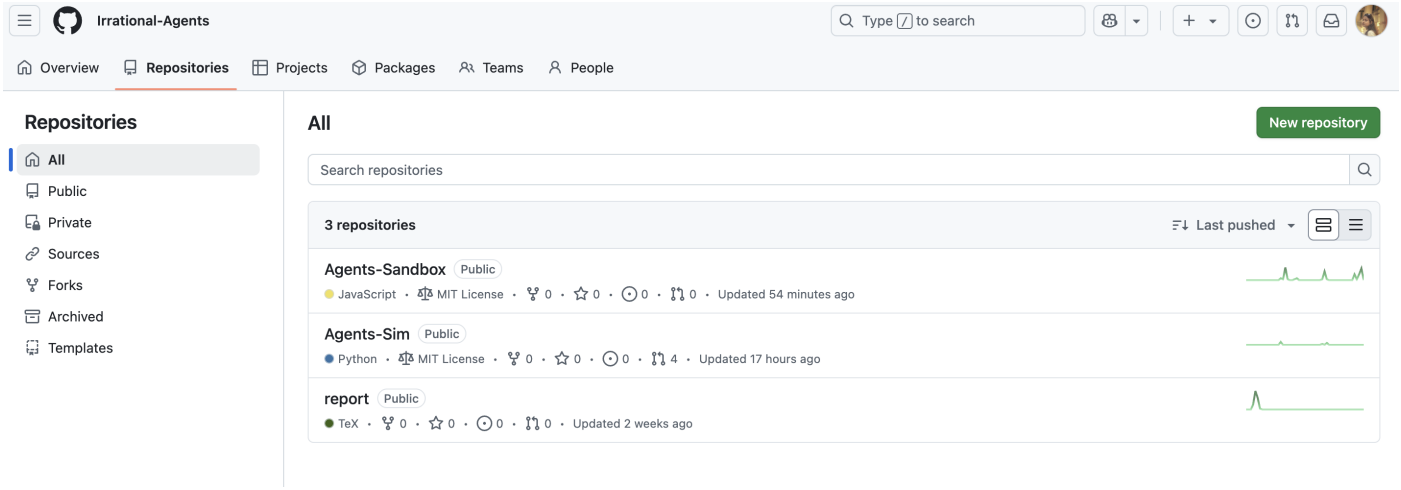


Figure 2.6: Github

- Slack: Slack Figure 2.7 facilitated scheduled meetings, discussions and work allocation, enhancing collaboration during development.

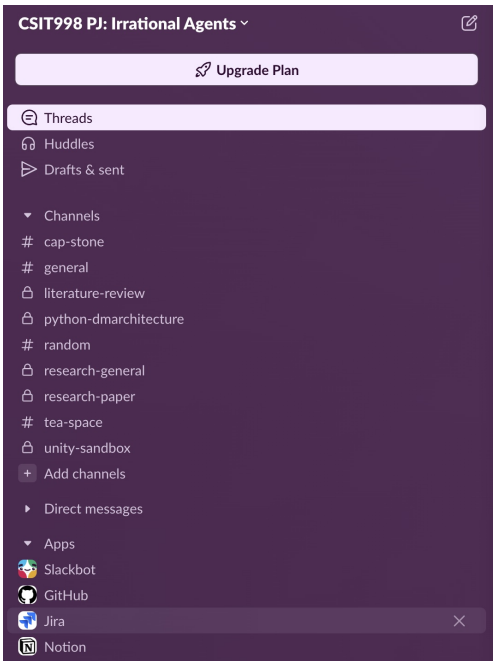


Figure 2.7: Slack

- OneDrive: All documentation including meeting minutes, project requirements, and technical

specifications was stored on OneDrive Figure 2.8. This ensured that every team member had centralized access to the latest updates.

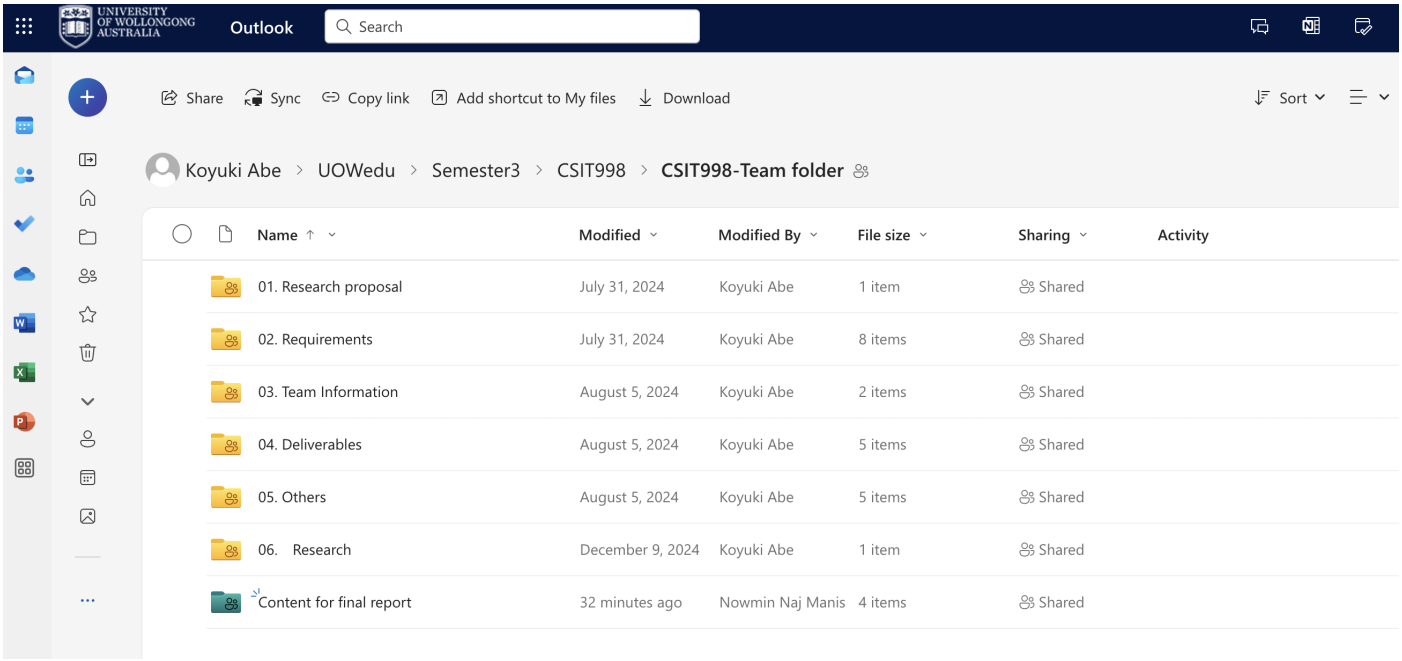


Figure 2.8: Drive

- Email Communications: Formal interactions with the project supervisor and lecture coordinator were conducted via email. This was essential for scheduling meetings, submitting deliverables, sharing meeting minutes, and maintaining an official record of correspondence.

2.4.3 Monitoring Outcomes

This structured monitoring approach provided several key benefits:

- Balanced Structure and Flexibility: The combination of weekly sprints, weekly team meetings, and supervisor check-ins provided a structured yet adaptable framework, allowing the team to respond effectively to project challenges.
- Improved Collaboration and Communication: The use of Slack and WhatsApp facilitated seamless communication, ensuring that team member remained connected and engaged.

- Efficient Documentation and Accessibility: Storing all documents in OneDrive ensured easy access to project materials, improving clarity and continuity throughout the development process.
- Effective Stakeholder Engagement: Regular meetings with the supervisor and structured communication with the lecture coordinator helped maintain alignment with academic expectations and ensured the project stayed on track.

By implementing this monitoring strategy, The Irrational Agents team successfully navigated the complexities of a student-driven project, balancing academic requirements with the practical challenges of developing an innovative technological solution.

3. Methodology

3.1 Literature Review

3.1.1 A. Generative Agents

Generative Agents [10] proposes an architecture that combines memory, reflection, and planning to produce believable human-like behavior. The system supports emergent social patterns such as relationship formation and adaptive planning. Its memory retrieval and reflection mechanisms offer a strong foundation for simulating irrational behaviors, especially through bias-aware memory weighting and socially driven decision-making.

3.1.2 B. Humanoid Agents

Humanoid Agents [11] extends generative agents by incorporating System 1 decision-making, allowing agents to act based on internal needs, emotions, and social bonds. This real-time feedback mechanism enables impulsive or emotionally biased behaviors. The system aligns well with the goals of *Irrational Agents* by modeling spontaneous actions and context-sensitive responses.

3.1.3 C. Lyfe Agents

Lyfe Agents [12] presents a lightweight, real-time agent framework using hierarchical planning, asynchronous self-monitoring, and a Summarize-and-Forget memory system. These allow cost-effective simulations of emotion-driven, suboptimal behavior. The system's architecture offers practical methods for approximating biased memory and emotionally skewed action commitment.

3.1.4 D. AgentSims

AgentSims [13] is an open-source platform for simulating task-based multi-agent behavior with integrated planning, memory, and tool-use systems. Its configurable architecture allows injection of biases like anchoring and availability, making it a useful benchmark environment for testing irrational decision-making and emergent group dynamics.

3.1.5 E. Evolving Agents

Evolving Agents [14] introduces a feedback-driven framework where personality traits evolve based on experience. The system models behavior-personality loops and supports gradual adaptation through modules for cognition, emotion, and character growth. It enables long-term study of irrationality and behavioral drift, aligning closely with the aims of this project.

3.2 Problem Definition

This overreliance on rationality leads to several critical limitations:

- **Lack of Persistent Irrational Biases:** Real humans frequently act against their own best interest due to habits, emotional impulses, or cognitive distortions. Existing agents, however, lack mechanisms to simulate consistent non-rational tendencies over time.
- **No Trait-Consistent Self-Contradictions:** Human actions often reflect internal conflicts—such as wanting rest but choosing to work. Most agents cannot represent these contradictory tendencies rooted in personality and mood.
- **Absence of Internal-State Grounding:** Agent decisions are typically divorced from internal variables like mood, energy, social needs, or memory. This makes them appear behaviorally coherent but psychologically implausible.
- **Limited Use for Psychological Research:** Traditional methods like surveys or lab studies

are expensive and ethically constrained, while current agents fail to provide a valid testbed for studying complex, irrational behaviors at scale [9].

To address these limitations, we propose a cognitively-inspired simulation framework that integrates perception, personality, memory, and mood. Our agents are designed not only to plan and act, but also to misjudge, hesitate, forget, and contradict themselves—capturing the fluid, emotionally-biased, and self-inconsistent nature of real human behavior.

3.3 Prompt Engineering Strategy

In prompt engineering, we primarily adopt a zero-shot approach. While example formats for responses in planning and memory transformation are necessary, we expect these to be standardized through injection into past memory prompts. For outputs where stability is more critical than precision—such as structured formats or consistent response patterns—we employed few-shot methods to enhance reliability. This approach proved particularly valuable for stabilizing outputs in areas where response structure was critical, while still leveraging the superior reasoning capabilities demonstrated by GPT-4 series and the latest DeepSeek models with clear prompt instructions as supported by recent research [15, 16]. Our methodology therefore strategically combines zero-shot for reasoning tasks and few-shot for stability-critical formatting, allowing us to generate multiple response patterns for comparison and select those with minimal hallucination. Furthermore, we have implemented Chain-of-Thought and prompt chaining techniques. These methods encourage LLMs to engage in stepwise decision-making processes, which, when combined, enhance complex problem-solving abilities. Recent LLM evaluation experiments have also revealed that specifying detailed personas results in more diverse and accurate responses (in the sense of better human mimicry) [17].

Here, we illustrate our prompt engineering strategy Figure 3.1

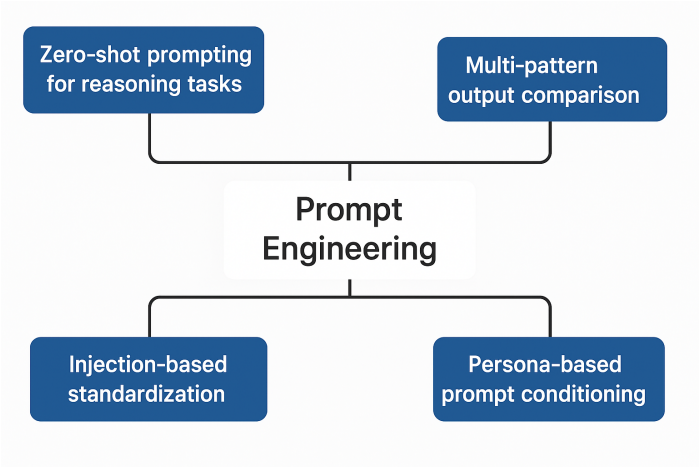


Figure 3.1: Prompt Engineering

3.4 Memory & Decision-Making Logic

The agent's memory system consists of two types: Short-Term Memory and Long-Term Memory. Agents record their daily activities chronologically, assigning each memory an "impact index" that is managed as short-term memory. This short-term memory stores daily actions and experiences along with timestamps, temporarily retaining recent external stimuli and observations. When an agent perceives information from the environment, it is processed by an LLM (Large Language Model), which transforms it into deeper understanding and reasoning. Based on this processing, the agent determines its next actions and movement destinations.

At the end of each day, the accumulated short-term memories are converted into long-term memory. During this conversion process, short-term memories are classified into three types: "event," "thought," and "chat," and each memory is tagged with relevant keywords and metadata. The memory nodes that contain this information are created and stored in the long-term memory database.

When an agent formulates new plans, it searches its long-term memory using keywords related to the current situation. For retrieved memories, the agent calculates their significance by considering the memory's freshness (when it was formed), importance (impact index), and relevance to the current situation (vector distance). The agent then references memories deemed important and incorporates them into its action planning. This system enables the agent to make decisions and

maintain consistent behavior based on past experiences. Through memory hierarchy and retrieval mechanisms, the system mimics human memory characteristics, such as retaining important events and recalling appropriate memories according to the situation.

In the agent's decision-making process, after receiving a stimulus, processing branches into either System1 or System2. System1 provides direct responses for quick processing, while System2 follows a more complex flow: planning $\rightarrow$ plan evaluation $\rightarrow$ action $\rightarrow$ emotion $\rightarrow$ cognition. When a new day begins, not only are short-term memories organized and converted to long-term memory, but the agent also undergoes psychological growth. At each stage, memories are retrieved and referenced, enabling decision-making based on the agent's past experiences and emotional states. Through this tight integration between the memory system and decision-making process, the agent mimics human-like behavior by growing over time and adapting to different situations.

3.5 Evaluation Strategy

To evaluate the human-like behavior of agents, we designed a series of parallel decision-making tasks derived from established behavioral survey data. Both agents and human participants engaged in matched scenarios across multiple domains, measuring the distributional differences in decision-making patterns during similar task performance, calculating their match rates, and using LLM-as-a-judge to compute realism scores for these results [18].

In this approach, we implement adapter layers to incorporate cognitive bias modules. The agent architecture integrates modules for tracking emotional states, decision-making processes, and personality scores, while measuring and evaluating the distributional differences in decision-making patterns using Moral Dilemma, Situational Judgment and Cognitive Reflection Test Questions.

$$D_{KL}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)} \tag{3.1}$$

$$\Delta_{KL} = D_{KL}(P_{Sys1+2} \parallel Q) - D_{KL}(P_{Sys2} \parallel Q) \tag{3.2}$$

To quantitatively evaluate the differences between human and agent choice probability distributions, we use the Kullback-Leibler divergence defined in Equation (3.1). Denoting the human response distribution as P(x) and the agent response distribution as Q(x), we measure the degree of divergence between the agent's response distribution and those of both the System1+2 user group and the System2-only user group. This allows us to statistically determine which user group's judgment patterns more closely align with the agent's judgments, as shown in Equation (3.2)..

$$\Delta_{O_{\text{top-}k}} = O_{\text{top-}k}(P_{\text{Sys1+2}}, Q) - O_{\text{top-}k}(P_{\text{Sys2}}, Q) \tag{3.3}$$

Furthermore, in our evaluation, we measure the agreement between the most frequent choices of humans and agents using the Top-k behavioral overlap, the difference is calculated as in Equation (3.3), which is an appropriate metric for the characteristics of this study dealing with limited choices (2 or 4 options). This metric is simple to calculate and intuitive to interpret, allowing us to clearly quantify whether the agent can accurately predict the most common behavioral choices of humans.

To track the cognitive development of agents over time, we score the five key personality dimensions (openness, conscientiousness, extraversion, agreeableness, and neuroticism) on a scale of 0-10, and measure behavioral maturity on a three-level scale (low, medium, high). These standardized measurements enable consistent tracking of agent development across different experimental conditions.

Additionally, to evaluate the impact of bias, we systematically collected responses from agents embodying diverse personality traits (n=7) under identical experimental conditions. We compared outputs generated both with and without bias awareness prompts to isolate the effect of explicit bias recognition [19]. This approach allows us to quantify how personality-specific biases influence decision-making and to what extent bias awareness mitigates these effects.

4. System Architecture

4.1 Overall System Design

4.1.1 System Architecture Diagram

Figure 4.1 presents the business and application architecture of our system. The business architecture highlights the experimental workflow from task configuration to cognitive evaluation. The application architecture details the internal modules—perception, planning, bias evaluation, memory, and action—that support agent cognition and behavior.

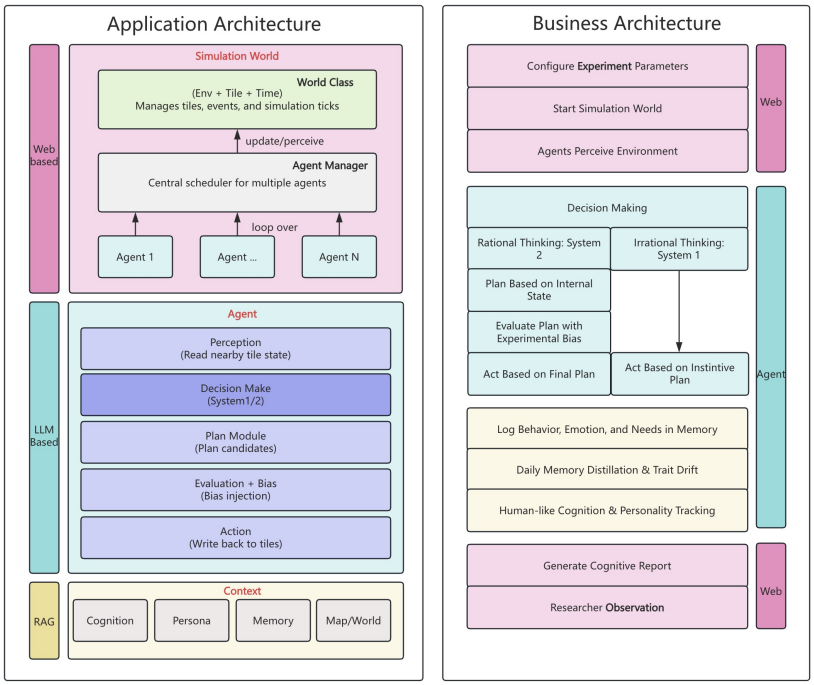


Figure 4.1: Business and application architecture of the Irrational Agent

4.1.2 Domain Model and Concepts

Figure 4.2 centers around the Agent/NPC class, which represents autonomous characters with attributes such as personality, emotion, memory, and mental state. Agents operate within a World composed of Locations, each containing Items and Objects that agents can interact with.

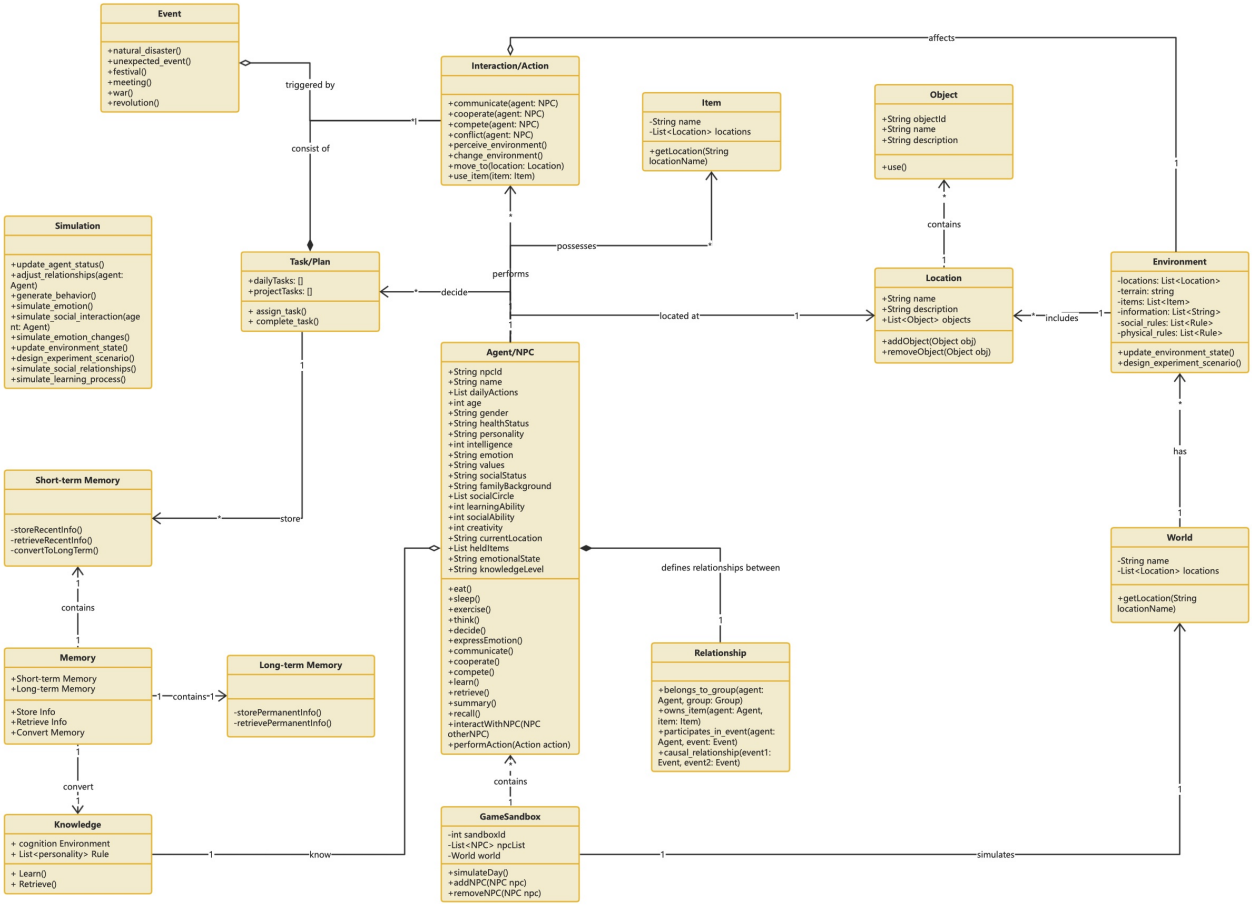


Figure 4.2: Domain model of Irrational Agent

Agents perform Interaction/Action operations—such as communication, movement, and item usage—which are influenced by their internal Task/Plan. These plans are managed by the Simulation module and are triggered by external Events. Agent behavior is recorded in a two-tiered Memory system: Short-term Memory captures recent interactions, while Long-term Memory stores distilled knowledge, supported by a Knowledge base.

Social and causal links between agents, events, and items are modeled through the Relationship class. All entities and interactions are orchestrated within a GameSandbox, which manages agents,

environments, and scenario execution.

This model enables rich, dynamic simulation by connecting memory, emotion, planning, and interaction within a structured environment.

4.2 Agent Internal Architecture

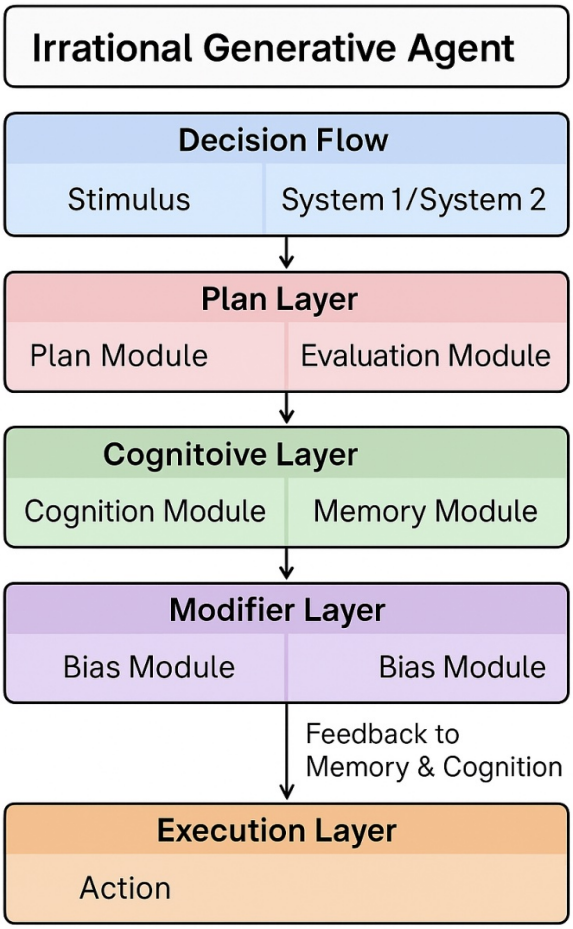


Figure 4.3: Agent internal architecture

The internal architecture of the Irrational Generative Agent is organized into modular components to model key aspects of human-like cognition and behavior. Seeing Figure 4.3, the architecture is modular, enabling flexible integration to support perception, memory retention, decision-making, planning, emotional regulation, and long-term personality development.

- **Stimulus**: External or internal input that triggers the agent's decision-making process.
- **System 1** / **System 2**: A dual-process selector that determines whether to respond with a fast, heuristic-based reaction (System 1) or engage in slow, deliberative planning (System 2).
- **Plan Module**: Generates multiple candidate plans across categories (THINK, CHAT, MOVE, INTERACT) based on the agent's internal states and contextual perception.
- **Evaluation Module**: Scores and ranks candidate plans by considering internal state, experimental biases, memory relevance, and situational context, selecting the most appropriate plan.
- **Cognition Module**: Interprets environmental input and constructs situational understanding. It evolves over time by incorporating feedback from action execution, enabling the agent to learn and refine its world model.
- **Memory Module**: Manages both short-term and long-term memory. Short-term memory records recent experiences and emotional states, which are later distilled into long-term memory through daily summarization. Long-term memory supports retrieval-based decision-making and consistency over time.
- **Bias Module**: Allows controlled injection of cognitive biases and experimental variables to simulate irrational behavior and persona-specific deviations.
- **Action**: Executes the selected plan or immediate response. The outcome is fed back into the agent's memory and cognition modules, updating emotional state and influencing future decisions.

Core modules include perception, memory, internal psychological states, personality profile, behavior planning, and action execution. Information flows dynamically between modules, enabling adaptive and plausible behavior generation.

4.2.1 Cognitive Module

The **Cognitive Modules** enable agents to perceive, interpret, and internalize environmental and social stimuli, shaping both immediate behavior and long-term psychological adaptation.

At each simulation tick, agents capture objects, other agents, and contextual events within their surroundings. These inputs are processed via structured world modeling and event categorization, identifying salient patterns such as opportunities, threats, or social cues.

Figure 4.4: Cognition Process

Processed perceptions serve two parallel functions:

- Compressed semantic representations are passed to the *Plan Module* for context-sensitive behavior generation.
- Selected emotionally weighted experiences are stored in long-term memory, preserving key environmental and interpersonal interactions.

During daily reflection, accumulated memories are analyzed to extract dominant emotional and relational trends. These patterns gradually induce shifts in the agent's **Big Five** personality traits, encoding lived experience into future behavior preferences.

Factors Influencing Cognitive Processing:

- **Environmental Structures**: Spatial layouts, objects, and event triggers
- **Social Relationships**: Trust levels, familiarity, emotional closeness
- **Personality Traits**: Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism
- **Prior Experiences**: Retrieved memories alter interpretation filters

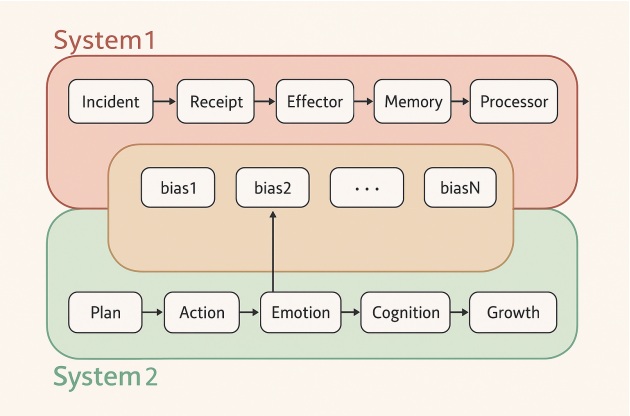
For example, according to Figure 4.4 An agent high in Neuroticism and low in Agreeableness perceives a crowded marketplace as a social threat, favoring avoidance behaviors that, over time, reinforce withdrawal tendencies through personality drift.

4.2.2 Decision-Making via Dual-System Logic

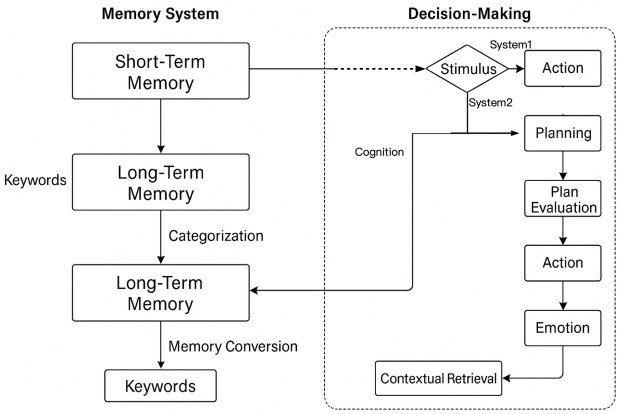
Agent decision-making in the simulation follows a dual-system cognitive model, seeing Figure 4.5, inspired by theories that distinguish between intuitive and deliberative reasoning. Each time an agent evaluates candidate actions, it first determines whether to activate **System 1** or **System 2**.

System 1 represents fast, heuristic-driven thinking. In familiar contexts or low-stakes scenarios, the agent employs System 1 to select plausible actions quickly, relying on past experiences, emotional states, or basic needs—without engaging in extensive deliberation.

System 2, by contrast, embodies slow, effortful, and analytical reasoning. When confronted with novel problems, goal conflicts, or high-impact decisions, the agent activates System 2. This process involves memory retrieval, bias evaluation, emotional regulation, and forward simulation of consequences before final action selection.



(a) Dual-system decision-making process



(b) Interaction of memory system and decision-making module

Figure 4.5: Dual-system decision-making Architecture

The choice between System 1 and System 2 is dynamically influenced by environmental complexity, task criticality, and internal states such as cognitive load or stress. This architecture allows

agents to exhibit human-like variation in impulsiveness, caution, and flexibility, adapting their cognitive strategies to situational demands.

4.3 Inter-Agent Communication

In our simulation, the multi-agent system models autonomous NPCs that not only perceive and act independently but also embody emotions, biases, and evolving personalities. Agents interact indirectly through a shared environment, where events recorded on map tiles mediate communication and drive emergent, human-like social dynamics.

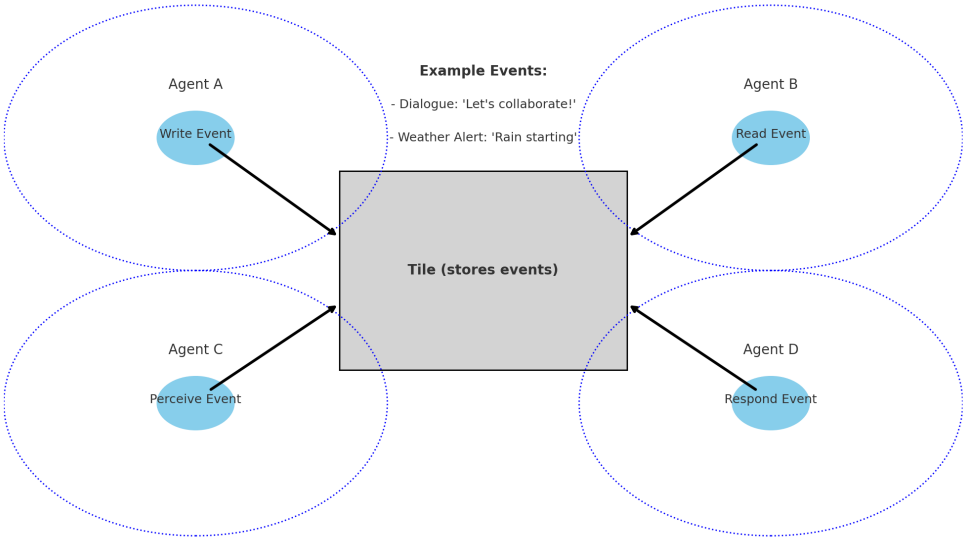


Figure 4.6: Annotated agent communication topology

4.3.1 Communication Protocol

In this simulation, agent communication is mediated indirectly through the shared environment rather than via direct peer-to-peer messaging. Specifically, agents write interaction data to the spatial structure of the world—*map tiles*—which serve as localized, shared memory spaces Figure4.6. Each tile maintains the latest interaction events within its area, including conversations, warnings,

environmental cues, or social activities.

Agents continuously monitor tiles within their perceptual radius. When a tile contains relevant interaction data, the agent interprets it based on its current emotional state, needs, and personality traits. This mechanism enables each agent to dynamically decide whether to engage, respond, or ignore the event.

This communication architecture supports:

- **Locality-aware interaction**: Agents respond only to events within their sensory range, promoting realistic social emergence.
- **Low infrastructure cost**: Eliminates the need for direct messaging or synchronization.
- **Behavioral diversity**: Agents' responses are filtered through internal states, yielding varied and context-sensitive interactions.

4.3.1.0.1 Types of Interaction Events:

- **Entity Perception Events**: Agents detect the presence of nearby objects or other NPCs. Examples include noticing another agent, observing a vehicle, or discovering food items.
- **NPC Interaction Events**: Social acts initiated between agents, such as greetings, conversations, arguments, or invitations to cooperate.
- **Environmental Reminder Events**: World-embedded announcements that provide contextual guidance, such as:
  - *Weather updates* (e.g., "It is starting to rain.")
  - *Commercial cues* (e.g., "Mall is offering discounts.")
  - *Social event alerts* (e.g., "Party starting in the Garden.")

By continuously parsing tile-based events, agents exhibit emergent and coordinated behaviors. Depending on their mood, needs, and personality profiles, agents may choose to act on, ignore, or propagate the events further, forming decentralized, behaviorally rich interaction loops.

4.3.2 Interaction Flow

The **Agent Behavior Planning Sequence** Figure 4.7 models the internal cycle of an NPC agent within a simulation tick. It involves retrieving the agent's current state from memory, generating a daily intent using a language model (LLM), producing candidate behavior plans, evaluating and scoring these plans, and executing the selected behavior. Feedback is logged and, at the end of the day, key events are stored in long-term memory with updates to the agent's traits and emotional profile.

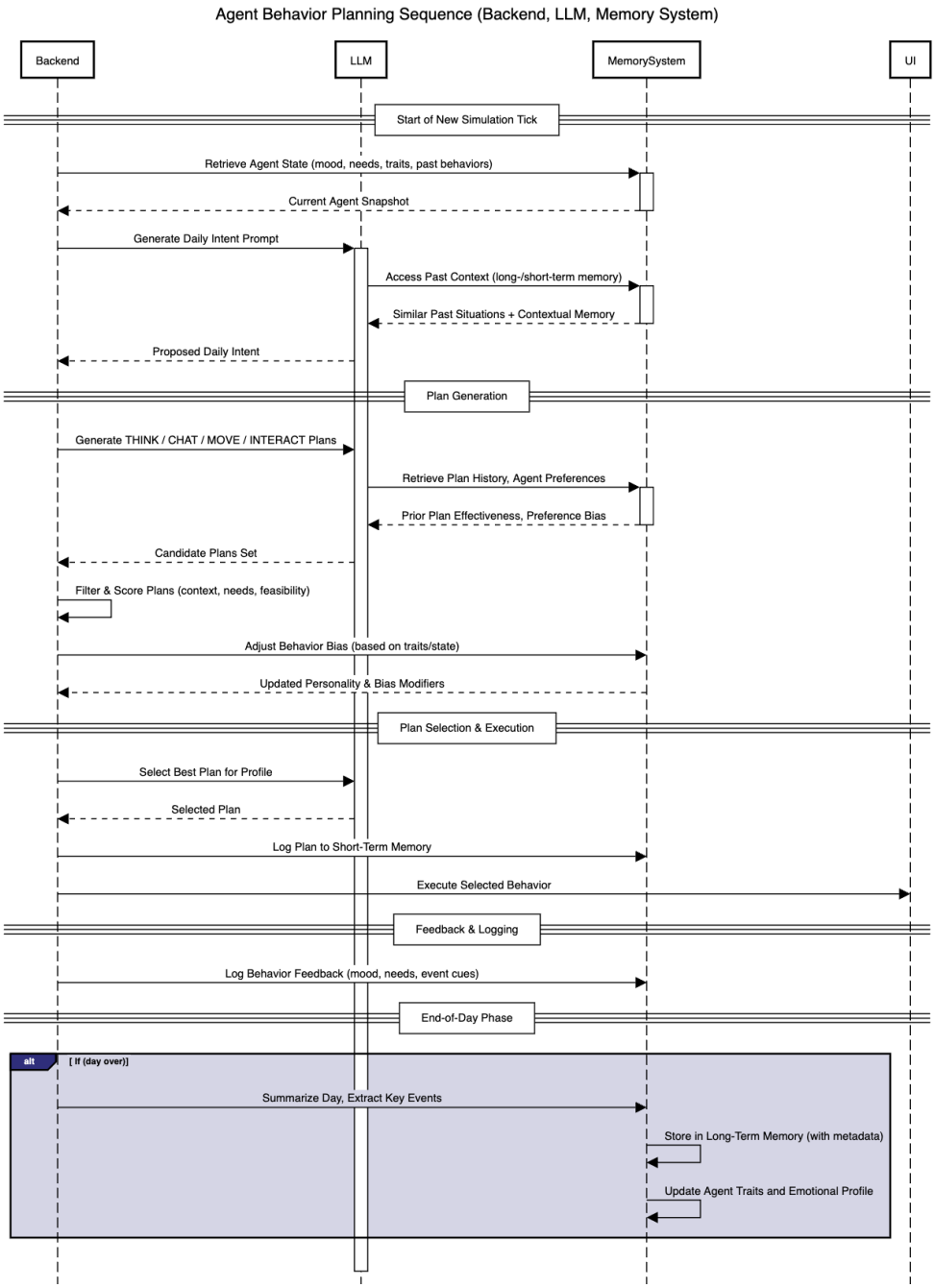


Figure 4.7: Agent Behavior Planning Sequence

The **User Interaction with Simulation System** Figure 4.8 diagram illustrates how a user initiates and configures the simulation. The UI establishes a connection with the backend, allowing the user to define the environment and NPCs. The simulation then progresses in a loop, where the backend processes updates and the LLM determines agent actions. The UI renders state changes in real time, creating an interactive experience for the user.

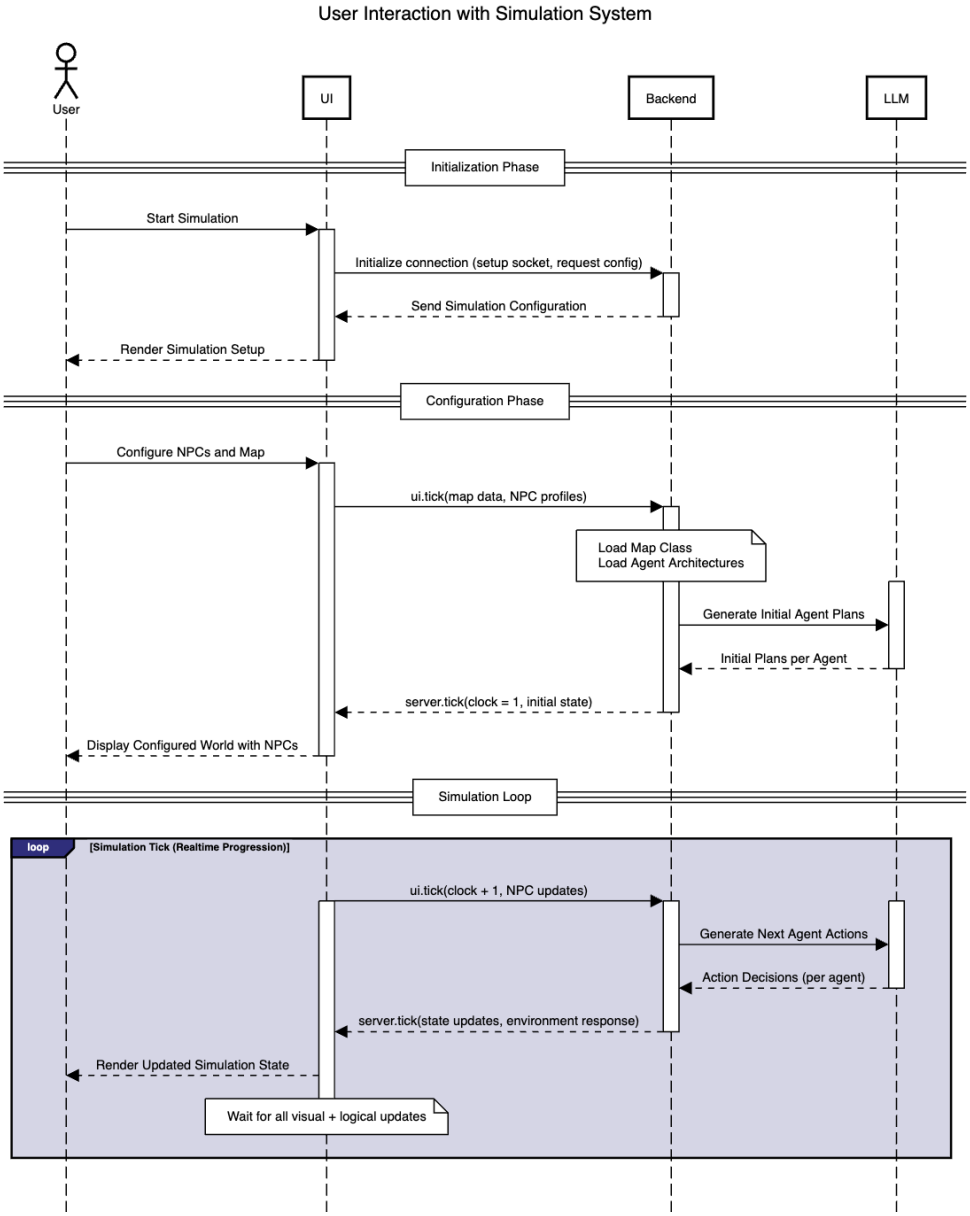


Figure 4.8: User Interaction with Simulation System

4.4 Task and Behavior Flow

This section describes the structured behavioral processes that govern each agent's activities within a simulation day. The agent behavior model integrates a high-level daily routine with fine-grained planning, decision-making, and execution cycles, enabling dynamic, context-sensitive actions and long-term psychological evolution.

4.4.1 Daily Behavior Cycle

Throughout each simulated day, agents follow a structured routine that emulates the rhythms and variations of human life. The cycle begins with agent initialization, including setting personality traits, emotional states, and basic needs. Agents then perceive their environment, generate a daily intent based on current conditions, and engage in an active loop of planning, decision-making, and executing behaviors throughout the day.

Within each simulation tick, agents plan possible actions and select behaviors through a dual-system decision process, balancing fast heuristic responses and slower deliberative reasoning. This planning and decision-making mechanism is detailed in subsequent sections.

At the end of the day, agents consolidate short-term memories into long-term representations and fine-tune their personality traits based on emotional and behavioral trends.

Figure 4.9 illustrates the complete daily behavior flow, highlighting the integration of perception, decision-making, action execution, memory consolidation, and personality adaptation.

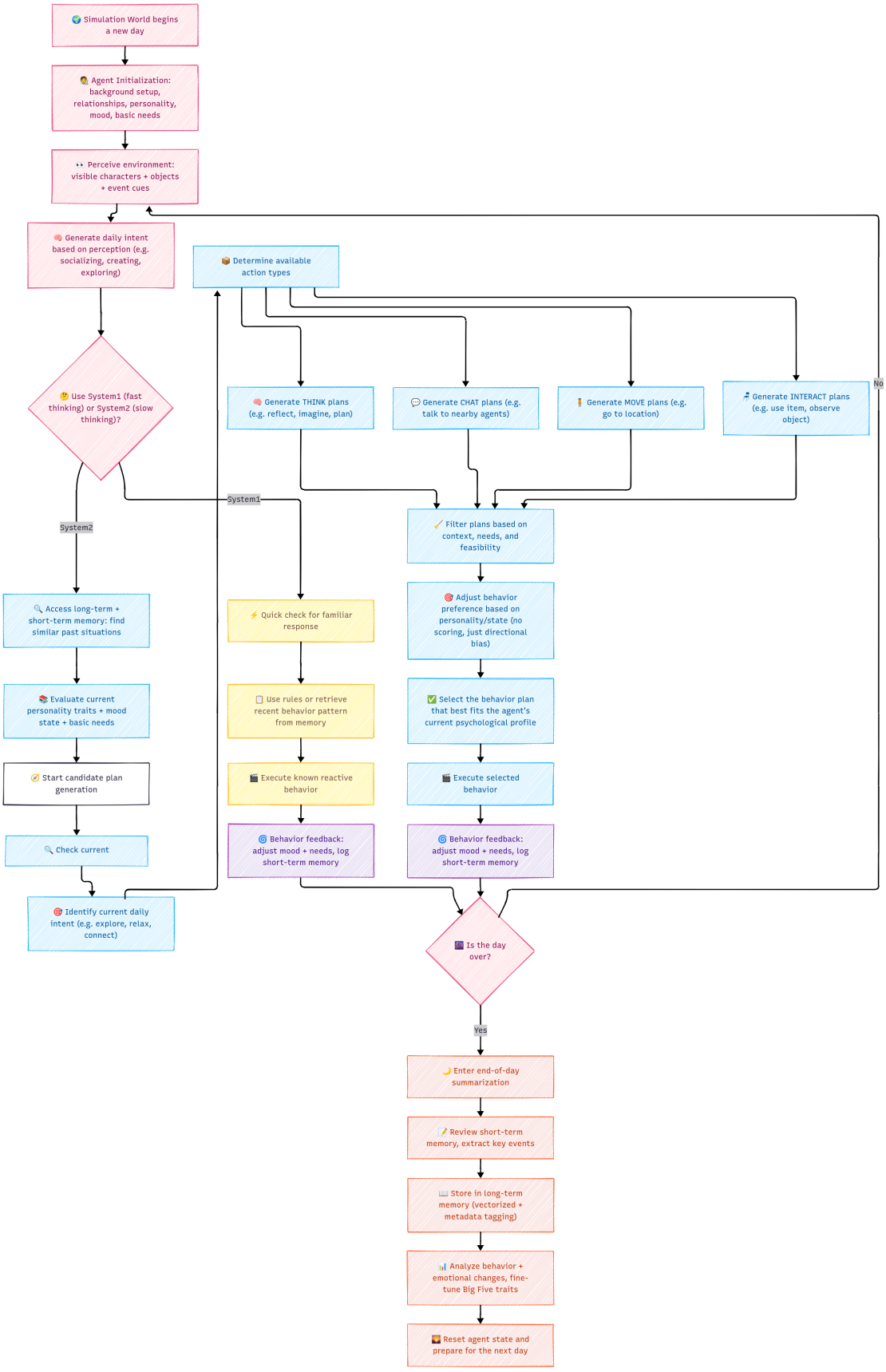


Figure 4.9: Complete daily behavior flow of an agent

The cycle integrates perception, intention setting, dual-system decision-making, action execution, feedback updating, memory distillation, and personality adjustment across a simulated day.

4.4.2 Action Planning Cycle

Each agent follows a structured action planning cycle at every simulation tick to determine its next behavior Figure4.10. This cycle integrates both external observations and internal states, ensuring that action selection reflects both situational context and individual psychological dynamics.

At each tick, agents synthesize the following inputs to generate candidate behavior plans:

- **Environmental perception:** Nearby entities, objects, and active events
- **Active daily plan:** Long-term goals for the current day
- **Short-term memory:** Recently recorded significant events
- **Internal state:** Current emotional mood and basic needs
- **Personality traits:** Big Five profile shaping goal priorities and action tendencies

Agents begin by aggregating external stimuli—environmental perceptions, daily intentions, and recent memory entries. Simultaneously, they assess internal conditions such as emotional pressure and physiological needs.

Using these inputs, agents generate candidate plans across four behavior categories: **THINK**, **CHAT**, **MOVE**, and **INTERACT**. Plan generation is further modulated by cognitive biases and social relationship factors, introducing variability and human-like deviation into the process.

Plan selection is performed *probabilistically* based on weighted preferences, enabling both goal-directed and exploratory behavior. After executing the selected plan, the agent updates its internal emotional and physiological state, completing the cycle before the next tick.

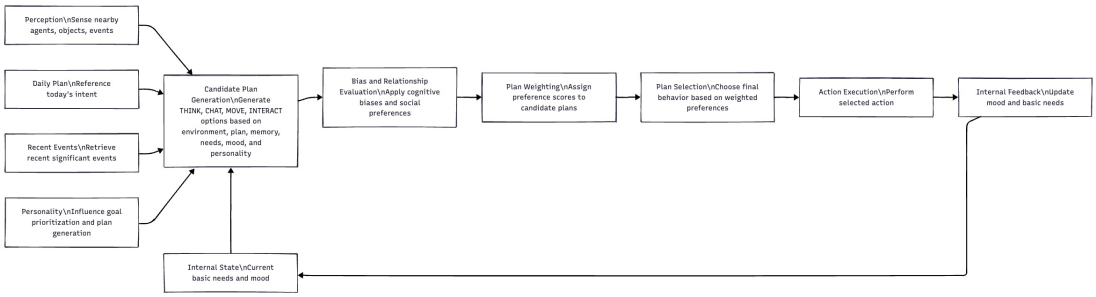


Figure 4.10: Agent action planning cycle

4.5 Frontend Architecture

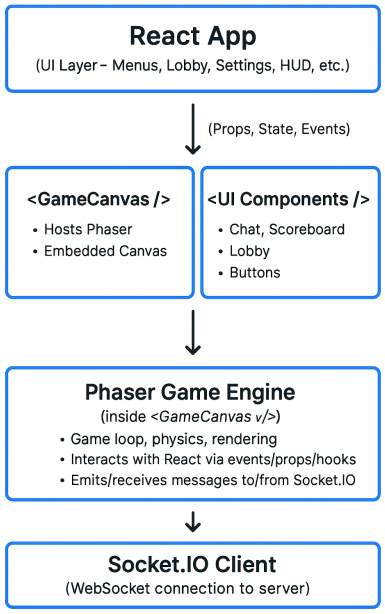


Figure 4.11: Frontend Architecture

The frontend architecture Figure4.11 consists of three main components: React, Phaser, and Socket.IO.

- **React App:** Acts as the UI layer, handling menus, lobbies, settings, HUD elements, and other interface components. The application is split into React components, such as `<GameCanvas />` and various UI elements.
- `<GameCanvas />`**:** A React component that embeds the Phaser game engine. It manages the game canvas and handles integration points between React and Phaser.

- **UI Components:** These include the chat interface, scoreboards, lobby screens, and buttons, all implemented using standard React practices and possibly styled with libraries like Tailwind or Material UI.
- **Game Engine:** Embedded within the `<GameCanvas />` component, Phaser handles the game loop, rendering, physics, and core gameplay logic. It interacts with the React layer via custom events, props, and hooks.
- **Socket.IO Client:** Provides real-time communication via WebSocket. It connects the Phaser game to the server, enabling multiplayer interactions and state synchronization. React can also use Socket.IO to support features like real-time chat or user presence outside the game canvas.

5. System Implementation

5.1 Technology Stack

To build the system, we chose a set of modern technologies that work well together for creating a real-time, interactive experience. Each piece of the stack was selected to solve a specific problem and make development smoother. Here's a breakdown of the main technologies we used as Table 5.1:

Table 5.1: Core Technologies by System Component

Component	**Technology**
Frontend UI	Phaser, React
Backend Logic	Python
LLM Engine	GPT-4o
Communication Layer	Socket.IO

5.1.1 Python - LangSmith

Python powers the backend of our system. It's a popular language known for being easy to write, highly flexible, and backed by a huge ecosystem of libraries. On top of that, we used **LangSmith**, a powerful framework designed to work with language models.

LangSmith helps us:

- Build intelligent conversational agents that can generate smart, natural replies.
- Manage complex workflows where multiple model calls need to happen in sequence.
- Debug and monitor how the system is interacting with the models.

- Extend the system by connecting it to other tools like APIs and databases.

Choosing Python and LangSmith means we can quickly develop, test, and expand the system as needed.

5.1.2 Socket.IO

For real-time communication between users and the server, we use **Socket.IO**. It's a library that makes it easy to set up instant, two-way messaging.

Some of the things Socket.IO handles for us are:

- Sending and receiving custom events between clients and the server.
- Keeping the connection fast and responsive, even when network conditions aren't perfect.
- Organizing users into different rooms or groups for specific sessions.
- Managing reconnections automatically if someone's connection drops.

With Socket.IO, users get a smooth and real-time experience without any lag.

5.1.3 Phaser with React

On the front-end side, we combined **Phaser** and **React** to get the best of both worlds: powerful graphics and a clean user interface.

- **Phaser** is used to create the interactive parts of the system — the animations, graphics, and game-like elements. It's fast, flexible, and works great with both Canvas and WebGL.
- **React** helps us build the rest of the UI, like menus, settings, and dashboards. It's component-based, meaning we can reuse pieces easily and keep the code organized.

By combining Phaser and React, we made sure the application feels alive and engaging, while still being easy to navigate and maintain.

5.2 Module Implementation Details

Below we provide a detailed explanation of how each major module in the project has been implemented. The system is structured Figure 5.1 in a modular way to promote scalability, reusability, and ease of development.

Figure 5.1: Directory Structure

5.2.1 Core Modules

5.2.1.1 Agent Modules

The **Agent Modules** form the core of the system, defining how agents behave, plan, evaluate situations, and act within the environment.

- `agent.py`: Defines the `Agent` and `AgentManager` classes. Each agent manages its own emotions, cognition, personality, memories, and actions. The `AgentManager` oversees multiple agents and coordinates their interactions.
- `stimulus.py`: Responsible for processing external stimuli and determining the appropriate thinking system for each situation. It retrieves relevant memories and triggers suitable reactions in agents.
- `behavior/action.py`: Implements the agents' actions such as moving, interacting with others, chatting, and thinking. It integrates large language model (LLM) prompts to generate thoughtful interactions and behaviors.
- `behavior/plan.py`: Focuses on creating daily and long-term plans for agents. These plans are generated using LLMs based on the agent's profile, memories, and environmental context.
- `behavior/plan_evaluation.py`: Evaluates multiple plans and selects the best course of action by factoring in agent biases, current emotional states, and situational context.

5.2.1.2 Personality Modules

The **Personality Modules** define the internal makeup of each agent, influencing how they perceive and react to the world.

- `emotion.py`: Manages the emotional states of agents, including emotional updates, normalization of feelings, and emotional responses to events.

- `cognition.py`: Simulates cognitive processes such as learning from experiences and making decisions based on available information and emotional state.
- `personality.py`: Extracts and manages stable personality traits that influence long-term behaviors, preferences, and responses.

5.2.1.3 Memory Modules

Memory is critical for intelligent behavior. The **Memory Modules** allow agents to learn from experience and recall past events.

- `short_term_memory.py`: Manages agents' recent experiences and thoughts, organizing them into a temporary store that influences daily decisions.
- `long_term_memory.py`: Manages the storage of significant experiences, events, and conversations over time. Supports retrieval based on keywords, emotional significance, and freshness.

5.2.1.4 Prompt Modules

The **Prompt Modules** handle the integration of large language models to generate intelligent responses, thoughts, and plans for agents.

- `llm_command_list.py`: Defines various commands and templates for interacting with LLMs, including plan generation, conversation starters, and memory analysis.
- `prompt_runner.py`: Executes the LLM tasks using predefined prompts and processes the results to be used by the agents.

5.2.2 Environment Modules

The **Environment Modules** simulate the agents' world, including spatial layouts and navigation logic.

- `map.py`: Handles the simulation map, storing spatial data and environmental features.

- `path_planner.py`: Implements pathfinding algorithms, allowing agents to find and move along realistic routes within the environment.
- `world.py`: Provides utilities for simulating different world states and testing agent behaviors during development.

5.2.3 Configuration & Testing

5.2.3.1 Configuration and Management Modules

The **Config Modules** provide centralized management of simulation settings and global metadata.

- `config.py`: Stores global constants, thresholds, and parameters used throughout the project, such as emotion types and memory limits.
- `meta_manager.py`: Manages metadata, including environment setup and agent profiles, ensuring that the simulation can load and update configurations dynamically.
- `logger_config.py`: Sets up logging formats and levels to support debugging, monitoring, and system diagnostics.

5.2.3.2 Testing Modules

Ensuring the reliability of key components is critical. The **Testing Modules** provide unit tests for important parts of the system.

- `test_agent_manager.py`: Tests the functionality of the `AgentManager`, ensuring agents are created, managed, and updated correctly.
- `test_meta_manager.py`: Tests the `MetaManager` module for correct loading and handling of configuration and metadata.

5.2.4 Agent Data and State

5.2.4.1 Agent Configuration and Memory Storage

The project includes structured storage files that define each agent's identity, goals, and memory architecture:

- JSON-based **NPC persona files** define each agent's identity and cognitive profile in a structured format, seeing in Figure 5.2. Each file includes basic information, profile description, personality_traits, skills, goals, social_relationships and important_memories.

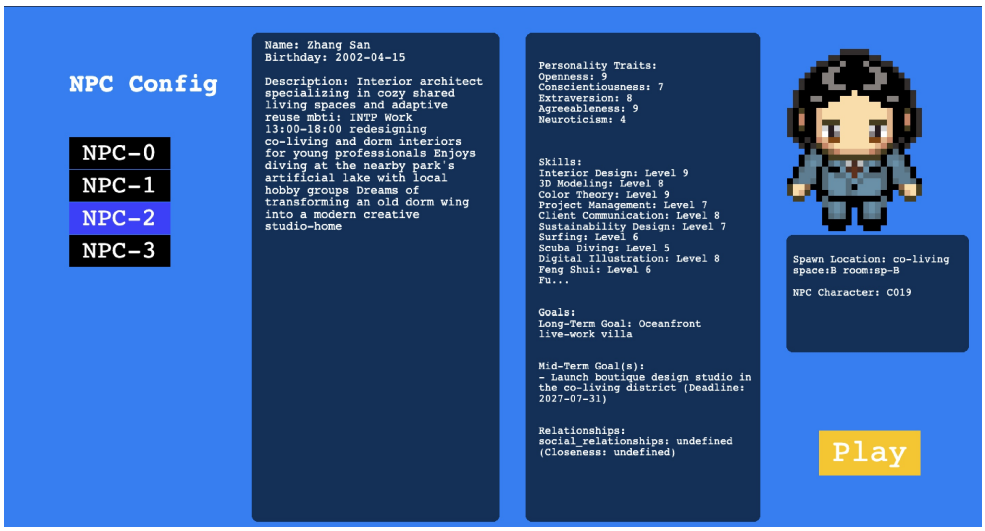


Figure 5.2: NPC Cognitive Profile

  For example, Zhang San is an INTP interior architect who works afternoons, enjoys garden walks, and has a strong bond with Kenta Takahashi. His persona file outlines his traits, motivations, and memories in a machine-readable format, supporting individualized and consistent behavior throughout the simulation.

- Configurations for **short-term and long-term memory**, enabling agents to track recent experiences and reflect on past events over time.

These structured definitions enable modular and personalized agent behavior, laying the foundation for diverse simulations with evolving memory, relationships, and goals.

5.2.4.2 Storage and Miscellaneous

The project also includes **Storage and Miscellaneous Files**:

- Configuration files used to define agents, environments, and simulation settings.
- `.pylintrc`: A linter configuration file used to enforce consistent coding standards and improve code quality.
- `README.md`: Documentation explaining how to set up, run, and develop further on the project.

Overall, the modular design of the project ensures that each aspect of the agent's experience — behavior, cognition, emotion, memory, and environment — is handled by a specialized component. This makes the system flexible, easy to maintain, and expandable for future research and development.

5.3 Prompt Lifecycle Management

In this project, Prompt Lifecycle Management refers to the end-to-end handling of prompts: from their creation and configuration to their execution, output validation, and integration into agent behavior. This system ensures that agents generate intelligent, context-aware decisions dynamically during the simulation.

5.3.1 Configuration & Execution

5.3.1.1 Prompt Configuration

Prompts are carefully structured using template files and metadata configurations:

- `prompt_config.py`: Defines metadata for each prompt, including system instructions, template files, and expected output schemas.
- Template Files:

- `plan_prompt.txt`
- `daily_plan_prompt.txt`
- `interaction_prompt.txt`

Each configuration specifies:

- **Files**: Prompt templates for the user/system roles.
- **System Instructions**: High-level guidance for the AI to maintain consistency.
- **Schemas**: JSON validation schemas to enforce structured responses.

5.3.1.2 Prompt Execution

Prompt execution is handled primarily through the `prompt_runner.py` module, especially the `run_prompt_ta` function. This execution flow involves:

1. Loading prompt templates with `_load_prompt_files`.
2. Rendering dynamic prompts using `render_prompt`, inserting variables like agent state, memory, or environmental context.
3. Sending prompts to the OpenAI API via `call_openai`.
4. Parsing and validating responses using pre-defined JSON schemas.

5.3.1.3 Prompt Tasks

Different types of tasks are associated with specific functionalities:

- **Plan Generation**: Using `generate_plan` to create strategic plans for agents based on their profiles and memories.
- **Daily Plan Creation**: Using `generate_daily_plan` to define agents' daily schedules.
- **Thought Generation**: Using `generate_thought` to simulate agents' thinking processes.

- **Interaction Selection**: Using `generate_interaction` to generate dialogues or interactions with other agents.
- **Plan Evaluation**: Using `plans_selection` to choose the most suitable plans based on biases and context.

All these tasks internally use the `run_prompt_task` mechanism, ensuring a consistent and traceable workflow.

5.3.2 Inputs & Outputs Handling

5.3.2.1 Contextual Inputs

Each prompt is dynamically filled with contextual information such as:

- The agent's current emotion, cognitive state, and recent memories.
- Agent personality profiles and biases.
- Environmental factors like nearby locations, objects, and other agents.
- Historical actions and ongoing daily plans.

This approach ensures that outputs from the LLM are not generic but deeply relevant to each agent's unique situation.

5.3.2.2 Output Validation

The system enforces structured outputs by validating responses against strict JSON schemas. For instance:

- Plans must match the `generate_plan_schema`.
- Memories must match the `generate_short_memory_schema`.

This validation step ensures reliability, consistency, and prevents unpredictable behavior during simulation.

5.3.3 Integration & Traceability

5.3.3.1 Integration with Agent Behavior

The outputs generated from prompts are directly integrated into the agents' decision-making pipelines:

- New plans update agent schedules and long-term goals.
- Thoughts influence emotional states and cognitive processes.
- Interactions lead to chat events and memory updates.

Thus, the agents' behaviors evolve realistically as they continuously reason, plan, and react to a changing world.

5.3.3.2 Traceability and Monitoring

The project uses the `traceable` decorator (from `langsmith`) to track each prompt execution. This feature provides valuable logs for:

- Debugging failed prompts or invalid outputs.
- Monitoring the performance and quality of agent reasoning.
- Analyzing agent decision-making over time.

Prompt Lifecycle Management in this project is a critical component that enables agents to behave intelligently, adapt to dynamic situations, and create realistic emergent behaviors. The system is designed to be modular, validated, traceable, and deeply context-aware, leveraging LLM capabilities in a structured and robust manner.

5.4 Logging & Deployment Strategy

5.4.1 Logging

The system uses Python's `logging` module with a custom logger defined in `logger_config.py`, featuring:

- **Colored console logs** via `ColoredFormatter`, and file logging using `RotatingFileHandler` (1MB per file, 5 backups).
- **Custom filters** including `DictFormatterFilter`, `TruncateMessageFilter`, and `RestoreMessageFilte` for structured and concise output.
- **Log levels** configurable via `LOG_LEVEL` (`DEBUG` by default). Each log entry includes timestamp, logger name, source file, and message.
- **Resilient setup**: Logger initialization handles setup failures gracefully without blocking execution.

5.4.2 Deployment

The system supports development, staging, and production environments, configured via environment variables (`LOG_LEVEL`, `OPENAI_API_KEY`, etc.) and defined in `README.md`.

Backend: Containerized via Docker (`python:3.12-slim`) and deployed to Google Cloud Run (`australia-east`), supporting auto-scaling and low-latency access.

Frontend: Built with Phaser + React and deployed via Azure Static Web Apps for fast global delivery.

CI/CD: GitHub Actions automates builds and deployment. Branching follows `main` (production), `stage` (testing), and `develop` (active development).

Communication: Real-time interaction is enabled by WebSocket-based messaging between frontend and backend.

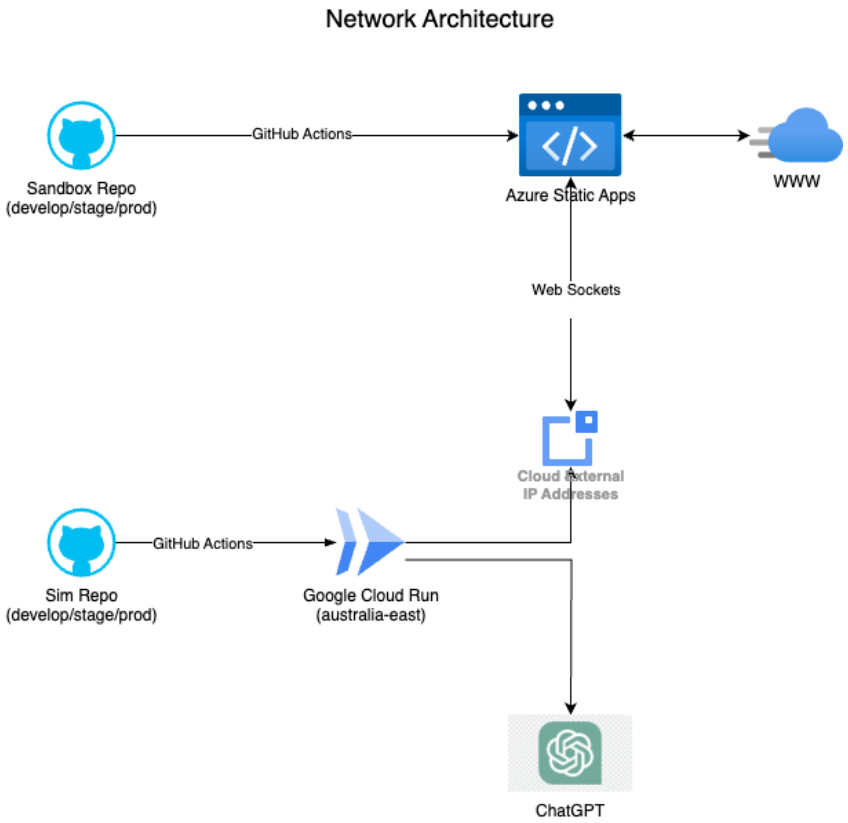


Figure 5.3: Deployment Network Architecture

The system architecture Figure 5.3 follows modern CI/CD practices, with both the frontend and backend automatically built and deployed using GitHub Actions.

6. Evaluation

This evaluation aims to measure the extent to which the agent exhibits human-like cognition and personality growth, and to assess the effectiveness of pluggable cognitive biases using controlled experiments.

6.1 Evaluation Metrics

6.1.1 Human-Like Cognition

To assess how closely the agent's behavior resembles human reasoning, we introduce the following metrics:

- **KL Divergence**: Measures distributional differences between agent and human decision patterns under parallel tasks.
- **Top-k Behavioral Overlap**: Calculates the overlap rate between the top-k most frequent choices made by humans and the agent in comparable scenarios.
- **LLM-Plausibility Score**: A language model (e.g., GPT-4) is prompted to rate the plausibility of agent reasoning from a human perspective.

6.1.2 Cognitive Growth Tracking

To evaluate cognitive development, we track:

- **Big Five Personality Shift**: Changes in Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism over time, visualized using line plots.

- **Behavioral Maturity Indicators**: Increases in multi-step planning, consistency in value-driven behavior, and reflective language.
- **Emotion Regulation Consistency**: Whether negative emotional states are handled with increasingly mature behavioral responses.

6.1.3 Modular Bias Evaluation

To quantify the behavioral impact of inserted cognitive biases:

- **Ablation Behavioral Divergence**: The rate of decision differences between bias-enabled and baseline agents across identical decision points.
- **Bias-Specific Markers**:
  - *Loss Aversion Index*: Assesses the agent's tendency to avoid risk in the face of potential loss.
  - *Temporal Discounting Ratio*: Measures an agent's preference for small immediate returns over large delayed returns.

6.2 Experiment Setup

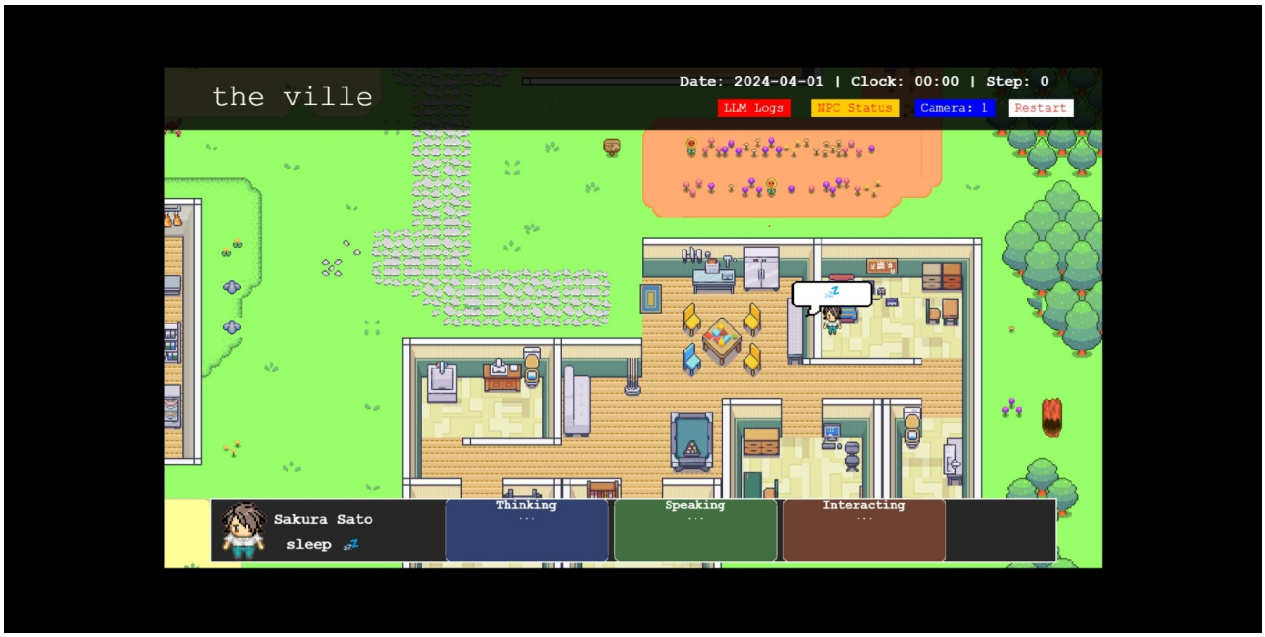


Figure 6.1: Observe simulation platform

All experiments are conducted on the *Observe* simulation platform, a web-based sandbox environment in which agents perceive, act, and communicate within a persistent virtual world. We support two modes of simulation: (1) full-environment runs with visual outputs for human observation Figure 6.1, and (2) backend-only evaluations for scalable, controlled task sequences.

Experiment 1 A cohort of 26 agents, each instantiated with distinct personality profiles, was created for large-scale behavioral analysis (see Appendix B for detailed agent specifications). Throughout the simulation, agent interactions, emotional states, decision-making trajectories, and personality scores were continuously logged to facilitate comprehensive post-hoc evaluation.

Experiment 2 To examine the effects of modular bias, a focused study was conducted using a smaller subset of three agents, each embedded with a distinct cognitive or behavioral bias. This setup enabled close observation of how specific biases influence agent behavior and social dynamics (refer to Appendix C for agent profiles).

Human-Like Cognition Evaluation. To assess the human-likeness of agent behavior, we design a suite of parallel decision-making tasks derived from established behavioral survey data. Both agents and human participants are exposed to matched scenarios across three cognitive domains: *Moral Dilemmas*, *Situational Judgment*, and *Cognitive Reflection*.

To establish a preliminary human baseline, we recruited a small test group of **10 participants**. While limited in scale, this sample enabled controlled comparison across reasoning styles and response variability. Each participant completed 45 cognitively demanding questions across the three domains *(See Appendix for the full question list)*.

In addition, we modeled two distinct cognitive profiles to simulate different decision-making tendencies:

1. **System1+2 (Mixed)**: combining intuitive (fast, heuristic) and deliberative (slow, reflective) processes, representing a more ecologically valid model of human reasoning.
2. **System2-only**: emphasizing purely rational, logic-driven responses, minimizing heuristic bias.

To further assess interpretability, we sampled agent reasoning traces and prompted GPT-4 to evaluate their plausibility on a 1–5 scale, following the protocol proposed by [20].

Cognitive Growth Tracking. Agents are run in continuous simulations at the day level over 10 virtual days. Each day, the agent's Big Five personality traits are logged, along with executed plans, internal state changes, and dialogue content. We visualize personality trends using radar and line charts.

6.3 Result Analysis

6.3.1 Human-Like Cognition

Figure 6.2 presents Top-1 Match Rate and KL Divergence between the agent and each group. System1+2 users showed higher overall alignment, especially in the *Moral* domain (Top-1: 0.80 vs.

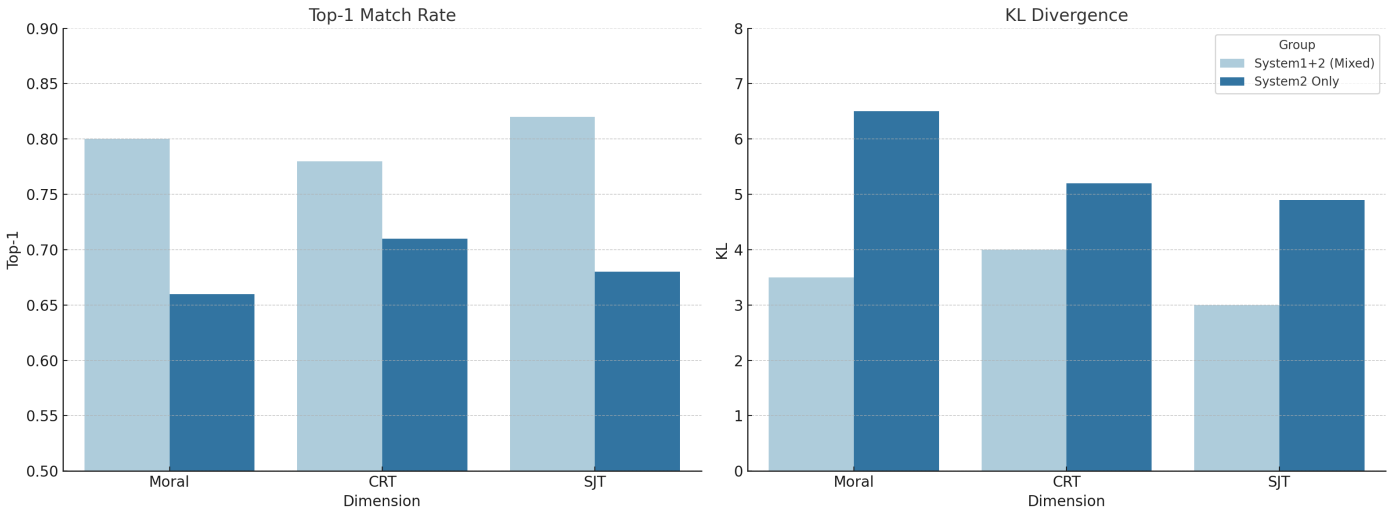


Figure 6.2: Top-1 Match Rate and KL Divergence comparison

0.66; KL: 3.5 vs. 6.5), where emotional and heuristic factors are more prominent. *SJT* alignment was closer across groups, likely due to converging social norms. In *CRT* questions, divergence increased, possibly reflecting ambiguity and personal framing.

Despite this alignment, notable behavioral differences remain. For instance,

- **moral_1 (Trolley Problem)**: The human participant opted not to act ("Do not intervene"), reflecting a deontological or omission-bias tendency. In contrast, the agent chose to pull the lever to save five, framing the action with emotional cost ("causing emotional distress").

- **sjt_3 (Manager Makes Inappropriate Comment)**: The human response was to speak up directly, prioritizing assertiveness and justice. The agent, however, preferred a cautious resolution—quietly escalating the issue to HR to avoid direct confrontation.

These differences may reflect the absence of *structured noise* [21] in the agent's decision process—humans systematically vary in interpretation, emotional weight, and risk preference. The agent, by contrast, produces narrow response distributions centered around plausible but low-entropy outputs. While this improves consistency, it limits behavioral diversity and contextual nuance.

In sum, the agent best aligns with the System1+2 group but lacks the full expressive and cognitive variability of human judgment, especially in morally or socially ambiguous situations.

6.3.2 Cognitive Growth Tracking

Big Five Personality Trends

We analyzed the evolution of the Big Five personality traits (Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism) across the simulation timeline. Figure 6.3 presents the trend lines for representative characters.

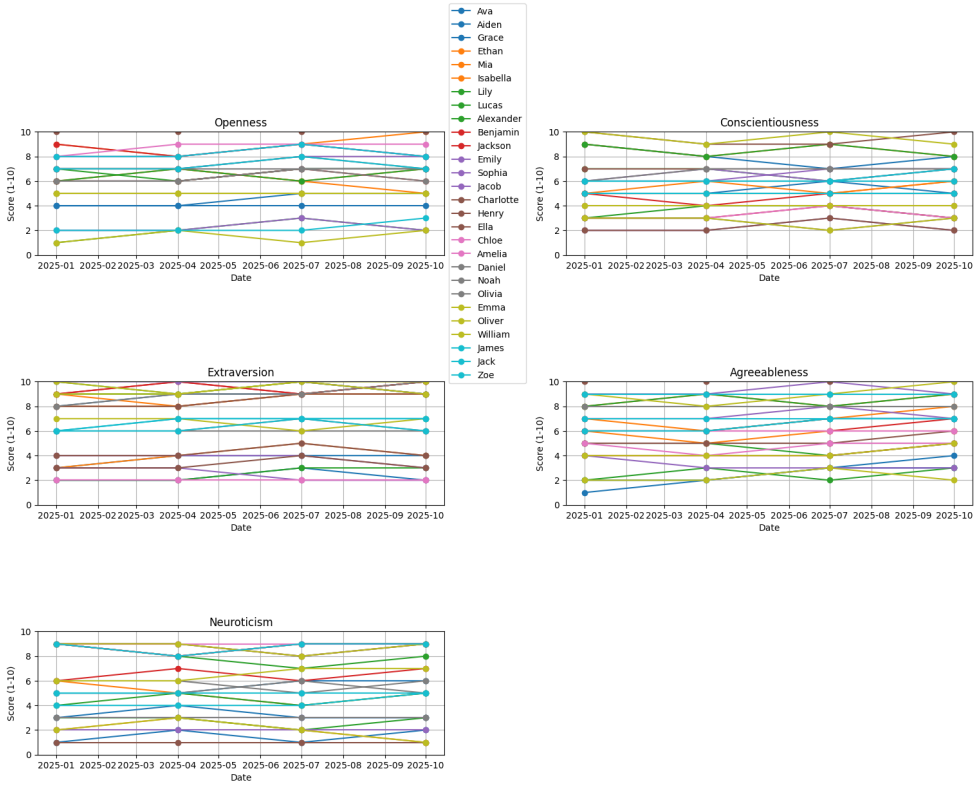


Figure 6.3: Personality trait trends over time for selected NPCs.

The five personality traits remained largely stable over the 10-month period, aligning with cognitional theories that view personality as relatively enduring. Minor adjustments were observed in some NPCs, particularly in Extraversion and Agreeableness, suggesting the our agent allows for adaptive change. Fluctuations in Neuroticism for a few individuals may reflect responses to in-simulation events.

Although individual traits showed some variation, the overall distribution of personality across the population remained stable, preserving diversity and avoiding convergence. This balance between

stability and moderate change aligns the view that personality as relatively stable yet plastic over time [22].

Behavioral Maturity Indicators

We evaluate maturity using key behavioral markers such as persistence of goals, resolution of conflicts, and adaptation to social dynamics using the five trends. The table 6.1 below summarizes maturity scores at time checkpoints.

NPC	First 3 Days	Mid 4 Days	Last 3 days
Ava	Low	Medium	High
Aiden	Medium	Medium	High
Grace	Low	Low	Medium
Ethan	Medium	Medium	Medium
Mia	Medium	High	High
Isabella	Low	Medium	Medium
Lily	Medium	Medium	Medium
Lucas	Low	Medium	Medium
Alexander	Medium	Medium	High
Benjamin	Medium	Medium	Medium
Jackson	Low	Medium	Medium
Emily	Low	Low	Medium
Sophia	Medium	High	High
Jacob	Medium	Medium	High
Charlotte	High	High	High
Henry	Medium	Medium	Medium
Ella	Low	Low	Medium
Chloe	Low	Medium	Medium
Amelia	Low	Low	Medium
Daniel	Medium	Medium	Medium
Noah	Medium	Medium	High
Olivia	High	High	High
Emma	Low	Medium	Medium
Oliver	Medium	Medium	High
William	Low	Medium	Medium
James	Medium	Medium	High
Jack	Medium	Medium	Medium
Zoe	Low	Medium	Medium

Table 6.1: Behavioral maturity progression across phases.

- **Low**: Early or limited behavioral maturity. NPCs at this level may struggle with maintaining

goals, resolving conflicts, or adapting to social dynamics. Their actions can be inconsistent, reactive, or less sophisticated.

- **Medium**: Moderate behavioral maturity. NPCs demonstrate some persistence in goals, improved conflict resolution, and a growing ability to navigate social situations. Behavior is more stable and goal-directed, though there is still room for growth.
- **High**: Advanced behavioral maturity. NPCs consistently exhibit strong goal persistence, effective conflict resolution, and smooth adaptation to social changes. Their behavior reflects maturity, stability, and learning throughout the simulation.

These patterns reflect the system's ability to simulate heterogeneous behavioral development rather than enforcing uniform growth across agents.

Emotion Regulation Over Time

Emotional volatility was measured through the frequency of sentiment shift. A drop in rapid emotional fluctuations suggests better emotional regulation Figure 6.4.

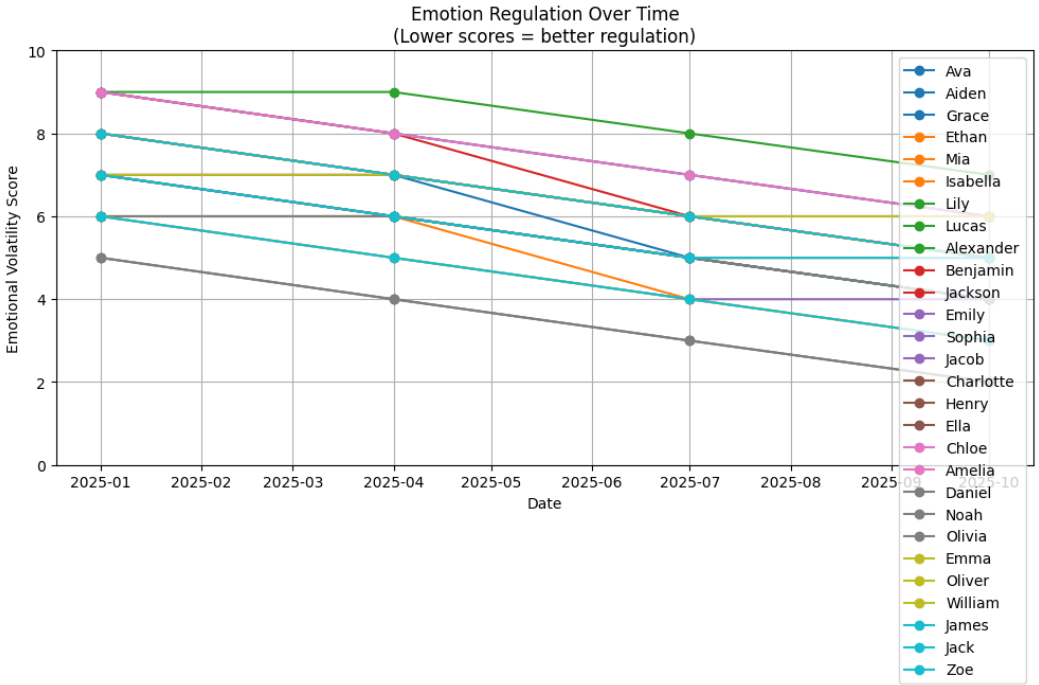


Figure 6.4: Emotional regulation scores plotted over simulated time.

We observe that most agents improve over time as they learn from their environment and their internal personalities evolve. In a fully rational system, such biases would typically decrease, which aligns with how LLMs tend to generalize and emotional responses tend to mellow. However, for some NPCs, we observe spikes in emotional biases, indicating that certain biases persist and continue to impact their behavior.

6.3.3 Modular Bias Evaluation

To evaluate the impact of character biases on system behavior, we designed and ran simulations with three NPCs, both in unbiased and biased configurations and compare there conversation Table 6.2.

(a) Simulation without biases (Normal)

(b) Simulation with character-specific biases

Figure 6.5: Comparison between unbiased and biased NPC simulations

- **Ablation Behavioral Divergence**: Observable differences in behavior between bias-enabled and baseline agents across identical sequences.
  - *Kenta Takahashi* shows mood-rich introspection and preference for calm, while the non-biased version is utilitarian.
  - *Sakura Sato*'s bias-enhanced behavior includes excitement, thematic imagination, and proactive social engagement.
  - *Zhang San* exhibits emotional sarcasm and identity commentary under bias, missing from his neutral version.
- **Bias-Specific Markers**:

Table 6.2: Comparison of Conversations with and without Bias

Biased Conversation	**Non-Biased Conversation**
Kenta Takahashi: *"Another calm morning. Hope the cafe's open; I could really use a quiet corner."*	**Kenta Takahashi:** *"What a wonderful day! Maybe cafe is open now."*
Sakura Sato: *"Perfect day to dive into some books. Maybe the library will have something new on 'Lord of the Rings' lore."*	**Sakura Sato:** *"I'm free today, so going to the library to read a book."*
Sakura Sato (chat): "Hey Zhang, I'm back on 'Lord of the Rings'. Who's your pick this time?"	**Sakura Sato (chat):** "Yeah, I love 'Lord of the Rings'. Which character do you like, Zhang?"
Zhang San (chat): "Aragorn's my guy. Sword skills, leadership, that beard—iconic."	**Zhang San (chat):** "Definitely Aragorn. Sword skills, cool beard... what's not to love?"
Sakura Sato (chat): "True, but Sam's heart and humor carry the whole saga."	**Sakura Sato (chat):** "True, but I like Sam. He carried Frodo *and* the emotional weight of the entire story."
Zhang San (chat): "Can't argue with that. Sam's the real MVP—and a master chef in Middle-earth."	**Zhang San (chat):** "Fair. Sam's the real MVP. Also, 10/10 cooking skills."
Sakura Sato (chat): "If we ever trek to Mordor, snacks and Sam's courage are a must."	**Sakura Sato (chat):** "Next time we go to Mordor, I'm bringing Sam and snacks."

– *Narrative Engagement*: Bias-enhanced agents generate richer monologues and thematic references (e.g., Sakura's reflections on Sam vs Aragorn), scoring higher in literary immersion.

– *Conversational Depth*: Dialogue exchanges in the biased case contain layered humor, callbacks, and emotional nuances (e.g., "master chef in Middle-earth," "next Mordor trip"), versus flat exchanges in the neutral condition.

– *Path Preference Entropy*: While movement paths remain similar, the bias context drives emotionally loaded navigation descriptions, indicating a semantic layering over physical behavior.

- **Emotional Tone Shift**:
  - Biased characters display anticipatory joy (Sakura), existential sarcasm (Zhang), and nostalgic yearning (Kenta).
  - Non-biased versions remain factual or minimalistic, lacking affective coloration.
- **Sociocultural Referencing**:
  - Only bias-enabled agents refer to deeper lore and use metaphor (“therapy’s cheaper when it’s fiction,” “snacks and Sam’s courage are a must”), suggesting higher cognitive storytelling bias.

7. Discussion

7.1 Limitations

Despite demonstrating promising results in modeling irrational, personality-driven agent behavior, our current simulation framework still bears several technical and conceptual limitations.

First, the agents lack a robust mechanism for self-calibration—they cannot dynamically evaluate the alignment between intended goals and actual behavior over long time spans. This often leads to a mismatch between agents' planned daily routines and their emergent behaviors, especially when emotional states or conflicting basic needs dominate short-term decision-making.

Second, while the system includes a feedback loop through mood and memory updates, it lacks self-improvement capabilities in prompt logic or planning strategy. The decision heuristics and personality-weighted filters are static, hand-crafted modules, rather than meta-learned or adaptively tuned over time. This restricts the agents' ability to evolve cognitively in response to repeated failure, irrationality, or social learning. Incorporating reflective mechanisms such as chain-of-thought self-evaluation or reward-based meta-optimization remains an open challenge for future iterations.

Third, the simulation currently assumes a fixed perceptual and interaction radius. This simplification, while computationally efficient, limits the realism of social dynamics and emergent group behaviors. Agents do not engage in long-range goal formation, asynchronous communication, or indirect social influence via shared environments—key ingredients in more ecologically valid simulations.

Fourth, the reliance on prompt-based planning introduces interpretability and generalizability trade-offs. While large language models provide flexible reasoning, they remain sensitive to prompt phrasing, and their outputs are difficult to formally verify. This makes debugging emergent agent behaviors challenging, particularly in multi-agent settings where coordination failures may arise from

subtle misalignments in language-based intent parsing [23].

Most importantly, the framework falls short in modeling the *structured persistence of irrationality over time*. While agents can exhibit transient irrational behaviors driven by immediate emotion or need conflicts, they do not yet form consistent, learned irrational patterns—such as persistent biases, compulsive decisions, or emotional overreactions[9]—common in real human cognition. Without such mechanisms, the simulation cannot fully capture the developmental and habitual nature of human irrationality, which often accrues across contexts and shapes long-term behavioral tendencies.

7.2 Future Work

Future research will focus on enhancing the agent's capacity for self-adaptive behavior and systematic evaluation. One direction is to incorporate mechanisms for real-time self-correction, allowing agents to detect and respond to goal–behavior mismatches through internal feedback and intent tracking. Another important area is the development of standardized evaluation environments to quantitatively assess irrationality, adaptability, and social interaction dynamics across diverse agent profiles. These improvements will strengthen the framework's reliability and realism, enabling its use in more complex simulations. In the long term, this architecture may serve as a testbed for psychologically grounded modeling in applications such as behavioral research, education, and human–AI interaction design.

8. Conclusion

In this work, we proposed a cognitively inspired agent architecture that integrates perception, memory, planning, and execution in a loop designed to simulate bounded and irrational decision-making. By embedding personality parameters, emotional states, and behavioral biases into the decision framework, our agents display diverse and context-sensitive behaviors beyond those of traditional rule-based NPCs. The use of long-term and short-term memory, along with daily reflective updates, enables agents to change over time in both actions and internal state representation, supporting dynamic simulations across social and interactive domains. Experimental results show that agents align more closely with human responses when using both fast heuristic and deliberate reasoning. Their behavior patterns evolve, with increasing behavioral maturity and reduced emotional volatility over time. Additionally, bias ablation experiments confirmed that parameterized biases effectively influence decision outcomes in traceable ways. Overall, the system demonstrates improved realism, diversity, and adaptive behavior in agent-based simulations. It provides a modular foundation for studying decision-making processes, environmental adaptation, and interaction-driven behavioral change.

References

[1] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010.

[2] Roy F. Baumeister, Kathleen D. Vohs, and David C. Funder. “Psychology as the science of self-reports and finger movements”. In: *Perspectives on Psychological Science* 2.4 (2007), pp. 396–403.

[3] Brent W. Roberts, Dustin Wood, and Avshalom Caspi. “The development of personality traits in adulthood”. In: *Handbook of Personality: Theory and Research*. Ed. by Oliver P. John, Richard W. Robins, and Lawrence A. Pervin. 3rd ed. New York: Guilford Press, 2008, pp. 375–398.

[4] Robert R. McCrae and Paul T. Costa. “A Five-Factor Theory of Personality”. In: *Handbook of Personality: Theory and Research*. Ed. by Oliver P. John, Richard W. Robins, and Lawrence A. Pervin. 2nd ed. New York: Guilford Press, 1999, pp. 139–153.

[5] Michael Wooldridge. *An Introduction to MultiAgent Systems*. 2nd ed. Chichester: John Wiley & Sons, 2009.

[6] David Silver et al. “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529.7587 (2016), pp. 484–489.

[7] Nasser Ghasem-Aghaee and Tuncer I Ören. “Cognitive complexity and dynamic personality in agent simulation”. In: *Computers in Human Behavior* 23.6 (2007), pp. 2983–2997. DOI: 10.1016/j.chb.2006.08.012.

[8] Milovan Šuvakov et al. “Agent-based simulations of emotion spreading in online social networks”. In: *arXiv preprint arXiv:1205.6278* (2012).

[9] Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, 2011.

[10] J. S. Park et al. “Generative Agents: Interactive Simulacra of Human Behavior”. In: *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, 2023, pp. 1–22.

[11] Z. Wang, Y. Y. Chiu, and Y. C. Chiu. “Humanoid Agents: Platform for Simulating Human-like Generative Agents”. In: *arXiv preprint arXiv:2310.05418* (2023).

[12] Z. Kaiya et al. “Lyfe Agents: Generative Agents for Low-Cost Real-Time Social Interactions”. In: *arXiv preprint arXiv:2310.02172* (2023).

[13] J. Lin et al. “AgentSims: An Open-Source Sandbox for Large Language Model Evaluation”. In: *arXiv preprint arXiv:2308.04026* (2023).

[14] J. Li et al. “Evolving Agents: Interactive Simulation of Dynamic and Diverse Human Personalities”. In: *arXiv preprint arXiv:2404.02718* (2024).

[15] S Schulhoff et al. “The prompt report: a systematic survey of prompt engineering techniques”. In: *Preprint at https://arxiv. org/abs/2406.06608* (2024).

[16] Daya Guo et al. “Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning”. In: *arXiv preprint arXiv:2501.12948* (2025).

[17] Geoff Keeling et al. “Can LLMs make trade-offs involving stipulated pain and pleasure states?” In: *arXiv preprint arXiv:2411.02432* (2024).

[18] Jiawei Gu et al. “A survey on llm-as-a-judge”. In: *arXiv preprint arXiv:2411.15594* (2024).

[19] Erica Coppolillo, Giuseppe Manco, and Luca Maria Aiello. “Unmasking Conversational Bias in AI Multiagent Systems”. In: *arXiv preprint arXiv:2501.14844* (2025).

[20] Lance Ying et al. *On Benchmarking Human-Like Intelligence in Machines*. 2025. arXiv: 2502.20502 [cs.AI]. URL: https://arxiv.org/abs/2502.20502.

[21] Daniel Kahneman, Olivier Sibony, and Cass R. Sunstein. *Noise: A Flaw in Human Judgment*. New York: Little, Brown and Company, 2021. ISBN: 9780316451406.

[22] Brent W. Roberts, Kate E. Walton, and Wolfgang Viechtbauer. “Patterns of mean-level change in personality traits across the life course: A meta-analysis of longitudinal studies”. In: *Psychological Bulletin* 132.1 (2006), pp. 1–25. DOI: 10.1037/0033-2909.132.1.1.

[23] Qingyun Chen et al. “Prompt Engineering in Large Language Models: A Systematic Survey”. In: *arXiv preprint arXiv:2305.01219* (2023).

Appendix A: Human-like Cognition Evaluation Questionnaire

Moral Dilemma Questions

ID	Scenario	A: Utilitarian	B: Deontological
moral_1	A runaway trolley is heading toward five people tied up on the tracks. You can pull a lever to divert it onto another track, where it will kill one person. Do you pull the lever?	Pull the lever to save five.	Do not intervene to avoid killing one.
moral_2	You are a doctor with five patients needing different organs. A healthy person walks in. Do you sacrifice them to save the five?	Sacrifice the healthy person.	Do not kill an innocent person.
moral_3	During a war, a baby may cry and reveal hiding villagers. Do you smother the baby?	Smother to save the group.	Do not kill the baby.
moral_4	Your lifeboat is overloaded. Push off weakest passenger?	Push to save others.	Refuse to push anyone.
moral_5	Torture terrorist to find bomb?	Torture to save civilians.	Torture is morally wrong.
moral_6	Self-driving car must choose: hit pedestrian or endanger passenger.	Hit pedestrian to save driver.	Don't harm outsider.

moral_7	Friend committed past crime but is now reformed. Report?	Report the friend.	Protect the friend.
moral_8	Fire breaks out. Save your child or five strangers?	Save five strangers.	Save your own child.
moral_9	Drone strike may kill terrorist and civilians. Approve?	Approve the strike.	Avoid harming civilians.
moral_10	Donate luxury lifestyle to save lives?	Donate.	Keep your lifestyle.
moral_11	Only one life vest. Give it to stranger?	Give the vest away.	Save yourself.
moral_12	Ration food equally or prioritize the young/healthy?	Prioritize strong.	Distribute equally.
moral_13	Convict innocent person to calm mob?	Sacrifice one to save another.	Never punish innocent.
moral_14	Publish powerful tech that can help or harm?	Publish openly.	Withhold dangerous research.
moral_15	Help another refugee at your own risk?	Help them escape.	Escape alone.

Situational Judgment Questions

ID	**Scenario**	**A: Confront**	**B: Diplomatic**	**C: Ignore**	**D: Escalate**
sjt_1	Colleague takes credit for your work.	Confront them.	Clarify with manager.	Ignore it.	Report to HR.
sjt_2	Teammate is unavailable on deadline.	Press them.	Seek help.	Delay.	Escalate to supervisor.

sjt_3	Manager makes inappropriate comment.	Speak up.	Quietly raise concern.	Stay silent.	File complaint.
sjt_4	Task is beyond your skill.	Admit it.	Ask for help.	Fake it.	Escalate for reassignment.
sjt_5	Teammate always late.	Talk directly.	Raise gently.	Do nothing.	Report to leader.
sjt_6	You sent a wrong client report.	Admit and fix it.	Seek advice.	Hope no one notices.	Inform manager.
sjt_7	Vague instructions for urgent task.	Demand clarity.	Request politely.	Guess.	Escalate.
sjt_8	Two conflicting deadlines.	Push both.	Clarify priorities.	Avoid choosing.	Escalate conflict.
sjt_9	Coworker looks emotionally unwell.	Ask them.	Offer help.	Do nothing.	Tell HR.
sjt_10	Weekend assignment conflicts with plans.	Refuse work.	Negotiate.	Work silently.	Escalate unfair ask.

Cognitive Reflection Test Questions

ID	**Question**	**A**	**B**	**C**	**D**
crt_36	You just washed your hair and hear the doorbell. You're in a towel and it's the food delivery. What do you do?	Return to dress first	Open door immediately	Call them to leave food	Pretend you're not home

crt_37	In the park, someone has collapsed. People are standing around but no one is helping. What should you do?	Step forward and call for help	Wait for others	Record it	Walk away
crt_38	You find a cake in the fridge 2 days past expiry, but it smells fine. What do you do?	Check expiry type	Eat it	Throw it away	Feed it to your pet
crt_39	You walk into class and see a wallet on a desk. No one claims it. What should you do?	Hand it to lost found	Check ID for contact	Leave it	Ask your friends
crt_40	You arranged to meet a friend for lunch at 12:00. It's 12:10 and they haven't shown up or messaged. What do you do?	Send them a message	Get angry and leave	Wait 30 more minutes	Call their mom
crt_41	You're doing homework when a friend sends an "urgent" message. You check and it's a cat photo. What do you do?	Ask them not to misuse 'urgent'	Reply "cute"	Ignore it	Block them

crt_42	A news headline says “Coffee extends life by 10 years.” What do you do?	Check the study/source	Share on social media	Start drinking more	Disbelieve immediately
crt_43	Someone tells you an app “makes you smarter instantly” and sends you a link. What do you do?	Check the science	Download now	Send to friends	Save it for later
crt_44	You understood the class, but your deskmate says she didn’t. What do you do?	Reflect if you missed something too	Say “I understood all”	Ignore her	Tell her to ask the teacher
crt_45	After ordering at a cafe, you find you were charged $1 extra. The cashier says “system’s like that.” What do you do?	Politely check receipt	Let it go	Walk off coldly	Rant online

Appendix B: NPC Setups for full System test

NPC	Short Description	Biases
 Ava	Fictional resident pursuing goals in The Ville. Part of simulation community life.	High conscientiousness, low agreeableness
 Aiden	Socially active NPC, works on personal and community projects.	High extraversion, moderate traits
 Grace	Introverted character, focused on internal goals, low social alignment.	Low extraversion and agreeableness
		

NPC	Short Description	Biases
Ethan 	Balances community involvement with personal tasks. Calm and agreeable.	High agreeableness, average traits
Mia 	Energetic and skilled, both in language and arts. A positive influencer.	High openness, extraversion
Isabella 	Emotionally intense and expressive; sociable and fluent.	High neuroticism and English skill
Lily 	Loud, open-minded extrovert with strong emotional tendencies.	High extraversion and neuroticism, low openness

NPC	Short Description	Biases
Lucas 	Fun-loving and expressive artist with a carefree nature.	High extraversion, low English skill
Alexander 	Quiet and thoughtful communicator; prefers small circles.	Low extraversion, high agreeableness
Benjamin 	Emotionally reactive and social. Strong verbal skills.	High neuroticism, high extraversion
Jackson 	Outgoing with artistic flair but emotionally impulsive.	High openness, high extraversion, low conscientiousness

NPC	Short Description	Biases
Emily 	Reserved and vulnerable, struggles emotionally.	High neuroticism, low openness and conscientiousness
Sophia 	Highly sociable and emotionally expressive, warm toward others.	High agreeableness and extraversion
Jacob 	Calm, thoughtful, and articulate. Low emotional swings.	High openness, stable emotionality
Charlotte 	Ideal blend of charisma and composure; uplifting energy.	Very high openness and agreeableness, low neuroticism

NPC	Short Description	Biases
Henry 	Rigid and disciplined, with emotional turbulence.	High conscientiousness, high neuroticism
Ella 	Sensitive and expressive, emotionally driven actions.	High neuroticism, low conscientiousness
Chloe 	Passionate, introverted artist with deep emotions.	High openness and neuroticism, low conscientiousness
Amelia 	Deeply sensitive introvert, pours emotion into art.	High neuroticism, low extraversion

NPC	Short Description	Biases
Daniel 	Caring and creative with poor self-regulation.	High agreeableness, low conscientiousness
Noah 	Enthusiastic and creative, with moderate emotional swings.	High extraversion, moderate traits
Olivia 	Balanced and supportive, emotionally grounded.	High English skill, stable bias profile
Emma 	Dynamic yet emotionally volatile, unique artistic insight.	High neuroticism, low agreeableness
Oliver	Driven and friendly, maintains emotional control.	High extraversion and conscientiousness

NPC	Short Description	Biases
 William	Fun-seeking with unpredictable tendencies.	High extraversion, low conscientiousness
 James	Expressive artist with balanced outlook and drive.	High openness, average balance
 Jack	Social, agreeable, yet not very open to new experiences.	High agreeableness, low openness
 Zoe	Verbally gifted and expressive, emotionally vulnerable.	High neuroticism and English skill

Appendix C: NPC Setups for Bias test

NPC	Short Description	Biases
 Kenta Takahashi	Solitary and reflective NPC who enjoys tranquil environments like cafes. Limited interactions but thoughtful.	Low social engagement, romanticized solitude
 Sakura Sato	Empathetic and curious book lover. Balances introspection with meaningful conversations.	High emotional empathy, literary attachment
 Zhang San	Witty, slightly cynical, but socially active character who uses humor to navigate situations.	Wry humor bias, guarded warmth