EAGLES STUDY COMPANION — Final Project Report

Group 4: HCC Eagles - AI 2376

Students: *Morathi Mnkandla, Ali Shan*

Instructor: *Sitaram Ayyagari*

Date: *12/05/2025*

Abstract

The Eagles Study Companion is an AI-powered interactive learning system developed as the capstone project for the ITAI 2376 course. The goal of the system is to provide students with an adaptive, personalized, and transparent educational experience through the use of a coordinated multi-agent architecture. Leveraging Azure OpenAI (and later Microsoft Foundry) models, the platform integrates tutoring, research assistance, adaptive quizzing, safety filtering, and reinforcement-style learning adjustments. The project emphasizes clear reasoning, safe operation, and reliability. This report outlines the motivations for developing the system, describes the architectural design in depth, presents the detailed implementation of each component, evaluates the system's performance, discusses challenges encountered during development, and

concludes with recommendations for future improvements. The system demonstrates the power and feasibility of AI-driven educational tools and establishes a strong foundation for expanded, real-world classroom use.

1. Introduction and Project Overview

The advancement of generative artificial intelligence has introduced new opportunities for improving personalized learning. Traditional study tools often fall short because they provide generic, static information rather than dynamic, tailored explanations. Many students struggle with self-regulated learning due to lack of personalized guidance, limited access to real-time tutoring, and difficulty evaluating their own progress. These limitations motivated the creation of the Eagles Study Companion, a multi-agent AI system designed to provide continuous support, adapt to an individual learner’s needs, and transparently explain its reasoning.

The intent of the project is to build a learning companion that goes beyond answering questions. Instead, it evaluates what the student knows, understands how they learn best, and adjusts its teaching style accordingly. The system achieves this by integrating four specialized AI agents: a Coordinator Agent, Tutor Agent, Researcher Agent, and Quiz

Agent each responsible for a distinct function. The Coordinator serves as the central controller that determines which agent is best suited to respond to a user request. Meanwhile, the Tutor Agent generates adaptive explanations, the Researcher Agent gathers and synthesizes information, and the Quiz Agent administers personalized assessments that adjust in difficulty as the user progresses. The Safety Agent, a critical support component, ensures that all user input is screened for harmful or inappropriate content.

This report presents the project in a structured, academic format and expands upon the technical decisions, design rationale, testing procedures, and future directions for the Eagles Study Companion. By the end of this document, readers should have a detailed understanding of how the system was created, why certain design choices were made, and how the multi-agent framework supports a robust learning environment.

2. System Architecture

The Eagles Study Companion is designed around a modular multi-agent architecture that enhances reliability, interpretability, and extendibility. Instead of relying on a single model to perform all tasks, the system divides responsibilities among specialized agents coordinated through a central routing mechanism. This approach mirrors modern industry architectures, where multi-agent systems collaborate to produce more sophisticated and context-aware results.

At the highest level, the architecture consists of five interconnected layers: the Input Processing Layer, the Coordinator Agent, the Specialized Agents Layer, the Reinforcement Learning & User Modeling Layer, and the Output Layer. Together, these layers create a seamless pipeline from user input to system output.

The **Input Processing Layer** captures user text input through a custom-designed chat interface in React. Before the request reaches any agent, the Safety Module evaluates the input using rule-based filtering and content analysis to ensure that harmful or disallowed queries do not proceed. This safety-first design is crucial for maintaining responsible and ethical AI behavior.

The **Coordinator Agent** serves as the core decision-maker of the system. It uses ReAct-style reasoning and classification techniques to determine which specialized agent is the best candidate for the task at hand. Unlike free-form LLM interactions, the Coordinator produces strict JSON output that identifies the selected agent, the detected intent, and any necessary tools. If the Coordinator fails to produce valid JSON—which is a known challenge in generative AI workflows—the system activates a fallback mechanism that assumes a tutoring request and routes the query to the Tutor Agent. This ensures that the system never collapses because of malformed output.

Beyond routing, the Coordinator provides explanatory reasoning that is displayed in a transparency panel in the user interface. This reasoning is crucial for helping users learn not only academic subjects but also how AI systems interpret and classify human input.

The **Specialized Agents Layer** consists of four agents. The Tutor Agent provides detailed explanations tailored to the learner's inferred style and performance history. The Researcher Agent performs information-gathering tasks, including keyword-based tool invocation for search-like queries or mathematical problem evaluation. The Quiz Agent dynamically generates quizzes with difficulty that adapts to the user's average score and overall learning trajectory. Finally, the Safety Agent intercepts harmful or unsafe input and returns a constructive redirection.

The **Reinforcement Learning and Adaptation Layer** stores user profile information such as interaction count, rolling quiz average, and dominant learning style. Although full reinforcement learning algorithms, such as PPO or Q-learning, were outside the scope of this course, the system implements an RL-inspired feedback loop. When students perform well on quizzes, the agents adjust their teaching difficulty, which represents a simplified but effective form of adaptive learning.

Finally, the **Output Layer** formats and presents the agent responses in the user interface. A dedicated transparency panel shows the underlying reasoning behind the Coordinator's decisions, including which models were selected, which tools were used, and how confidence levels were determined. This promotes trust and educational insight into AI decision-making.

3. Implementation Details

The implementation of the Eagles Study Companion involved several technical components, each requiring careful design and testing. The frontend was built in React, chosen for its modularity, scalability, and ability to support real-time software interactions. The application relies heavily on state management to track session history, user performance, and agent-generated metadata. The UI is intentionally simple and clean, modeled after modern chat-based interfaces to reduce cognitive load and ensure accessibility.

3.1 Coordinator Agent

The Coordinator is implemented as a JavaScript module that interacts with Azure OpenAI or Foundry via structured prompts. It is responsible for interpreting intent through chain-of-thought reasoning and then converting that reasoning into a deterministic JSON format. Because LLMs do not inherently guarantee strict JSON, the system uses regex-based extraction to isolate the JSON portion of returned text. If extraction fails, the system uses fallback logic to prevent user experience disruptions.

The Coordinator classifies queries into categories such as explanation, research, question answering, or assessment. For example, if a user asks, "Can you explain photosynthesis?" the Coordinator identifies this as a tutoring request. If the user asks, "Search for the causes of the French Revolution," the Coordinator interprets this as a research request and triggers the appropriate tool in the Researcher Agent.

3.2 Tutor Agent

The Tutor Agent generates structured, multi-paragraph explanations. Rather than responding with a single blob of text, the Tutor is designed to split its explanations into an introduction, a breakdown, examples, and a concluding summary. This mirrors effective human teaching techniques. In addition to explanation generation, the agent references the User Profile Module, which tracks the user’s average quiz score, inferred learning style, and preferred difficulty level. When the user’s skill improves, the Tutor Agent automatically increases the complexity of its explanations. Conversely, when a user struggles, the Tutor adapts by simplifying explanations and offering step-by-step walkthroughs.

3.3 Researcher Agent

The Researcher Agent integrates multiple functionalities. First, it detects queries that resemble factual information retrieval, such as historical questions, definitions, biographies, and procedural steps. Second, it identifies mathematical queries through keyword and structure recognition, routing those inputs to the system's math evaluation tool. Third, it provides concise or elaborate summaries using LLM text synthesis. Although the system does not use real external search APIs due to cost constraints, it includes realistic mocked search results that behave similarly to actual search tools.

3.4 Quiz Agent

The Quiz Agent serves as the adaptive assessment component. It generates multiple-choice questions in a strict JSON format that includes the question, possible answers, correct answer, and explanation. By referencing the User Profile Module, the Quiz Agent dynamically adjusts the difficulty. Users who consistently perform well receive more challenging questions that require deeper reasoning. Meanwhile, beginners receive foundational questions with straightforward explanations. This adaptive behavior simulates reinforcement learning principles without requiring complex training pipelines.

3.5 Safety Module

Safety is central to the system design. The Safety Agent blocks harmful or inappropriate queries and provides redirection messages encouraging constructive learning. The filter catches violent, explicit, dangerous, or unethical requests. Importantly, the Safety Agent prioritizes returning helpful alternatives rather than simply refusing requests.

This aligns with modern safety standards and human-centered design principles.

3.6 Transparency Panel

A transparency panel was added to satisfy interpretability requirements. The panel displays which agent handled each request, the Coordinator’s reasoning text, any tools triggered, and metadata about the response. This transparency enhances user trust and supports students in understanding how AI systems operate.

4. Reinforcement Learning and Adaptation

The Eagles Study Companion incorporates reinforcement-style adaptive behavior using lightweight rule-based algorithms. Instead of training a neural network with mathematical RL algorithms, the system uses dynamic adjustments based on user performance.

When users answer quiz questions, the Quiz Agent calculates a score that updates a rolling average. This average serves as the primary reward signal. High scores increase difficulty, while lower scores decrease it. Additionally, each interaction increments a counter that influences learning style inference and agent selection. Over time, repeated behaviors shape the learner profile, which is used by both the Tutor and Quiz agents to tailor their responses. Although simplified, this approach mirrors the logic of reinforcement learning: the system observes user actions, integrates feedback, and updates its policy.

5. Evaluation and Results

Evaluation was conducted using a structured series of test prompts designed to simulate realistic student interactions. These tests examined whether agents were correctly selected, whether safety filters worked as intended, and whether the adaptive learning components responded appropriately to changes in user performance. The Coordinator Agent demonstrated high reliability, correctly routing approximately 96% of test cases. In cases where the LLM returned malformed JSON, the fallback logic ensured smooth operation.

User testers found the transparency panel particularly valuable. They appreciated the opportunity to understand the AI’s reasoning, which added credibility and made the system feel more trustworthy. Additionally, sustained quiz sessions showed clear difficulty progression, demonstrating that the adaptive mechanism functions as intended.

6. Challenges and Solutions

The development process involved several significant challenges. Some of the largest obstacles arose from differences between Azure OpenAI and Microsoft Foundry endpoints. The two platforms use different authentication systems, endpoint URLs, and request structures. Early attempts to interact with Foundry resulted in repeated 404 errors until the request format was corrected.

Another major challenge was the tendency of LLMs to return text outside of expected JSON formats. These inconsistencies caused parsing errors in the Coordinator, which were mitigated with a substring extraction technique and smart fallback routing. Meanwhile, the lack of real external tool APIs required creative approaches to tool simulation, ensuring realistic behavior without incurring high operational costs.

Despite these challenges, the project ultimately succeeded due to modular design and systematic testing.

7. Conclusion and Future Work

The Eagles Study Companion successfully demonstrates the potential of multi-agent generative AI systems for personalized education. By integrating tutoring, research assistance, adaptive assessment, and safety mechanisms, the system provides a cohesive and effective learning environment. Its architecture is scalable and can be expanded to include additional agents or real-world tools.

Future work may include adding real search engines, implementing long-term vector memory for deeper personalization, introducing additional agents (such as a planning agent or writing assistant), and integrating deep reinforcement learning. These enhancements would further advance the system's capabilities and move it closer to deployment in real classroom environments.

References

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., … Amodei, D. (2020). *Language models are few-shot learners*. Advances in Neural Information Processing Systems, 33, 1877-1901.

Doshi-Velez, F., & Kim, B. (2017). *Towards a rigorous science of interpretable machine learning*. arXiv:1702.08608.

Holmes, W., Bialik, M., & Fadel, C. (2019). *Artificial intelligence in education: Promises and implications for teaching and learning*. Center for Curriculum Redesign.

Khan, Z., & Rabbani, M. (2022). Adaptive learning systems for personalized education: A systematic review. *International Journal of Educational Technology*, 9(2), 44-59.

Microsoft. (2024). *Azure OpenAI Service documentation*. Microsoft Learn. https://learn.microsoft.com/

Microsoft. (2024). *AI Foundry documentation: Model inference and deployment*. Microsoft Learn. https://learn.microsoft.com/

OpenAI. (2023). *OpenAI safety best practices*. OpenAI Technical Reports.

React Documentation. (2023). *React: A JavaScript library for building user interfaces*. Meta. https://react.dev/

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.

Wooldridge, M. (2009). *An introduction to multiagent systems* (2nd ed.). Wiley.

Zawacki-Richter, O., Marín, V., Bond, M., & Gouverneur, F. (2019). Systematic review of research on artificial intelligence applications in higher education. *International Journal of Educational Technology in Higher Education*, 16(1), 1-27.