Gemini CLI

Website

Link to Video

Presented by
Michael

Group Members

Ming Yuan

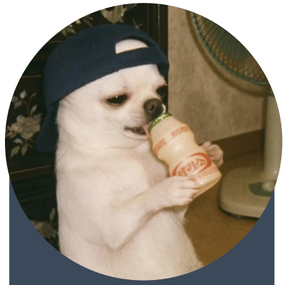
Team Lead + Main & Alternative Architecture Style + Overview + AI

22xb63@queensu.ca

Michael Liu

Presenter + Concurrency & Control Flow + AI

22jdn2@queensu.ca



Enrong Pan

Presenter + Subsystems Functionalities + Lesson Learnt + Derivation

enrong.pan@queensu.ca

Omar

Use case + External Interfaces

22chm1@queensu.ca

Kosi Amobi-Oleka

Abstract + Intro + Conclusion

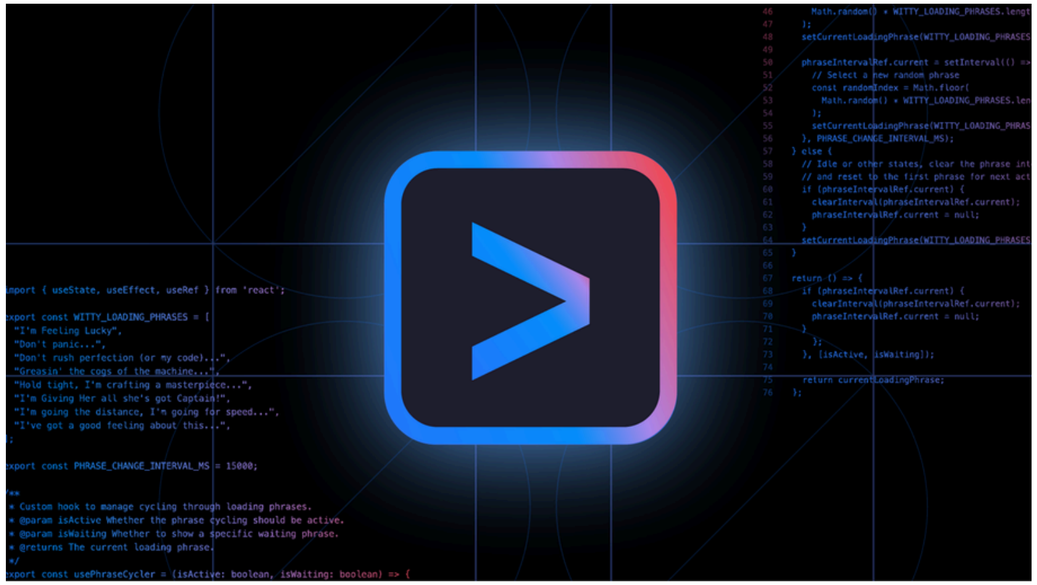
22fnbn@queensu.ca

Nandan Bhut

Subsystem Interaction + Intro

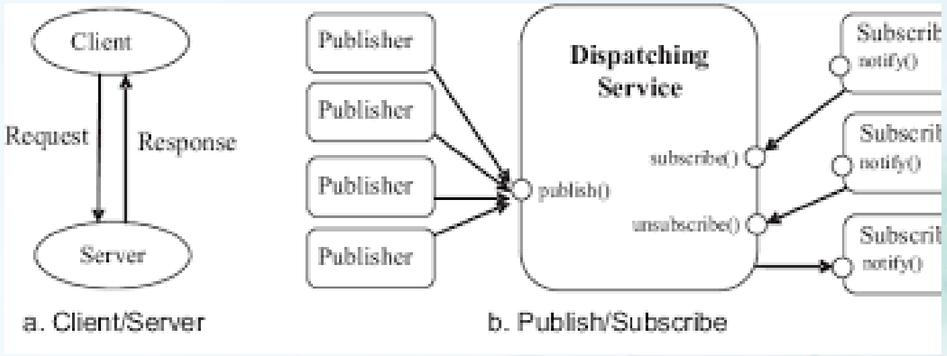
nandanbhut01@gmail.com

What is Gemini CLI

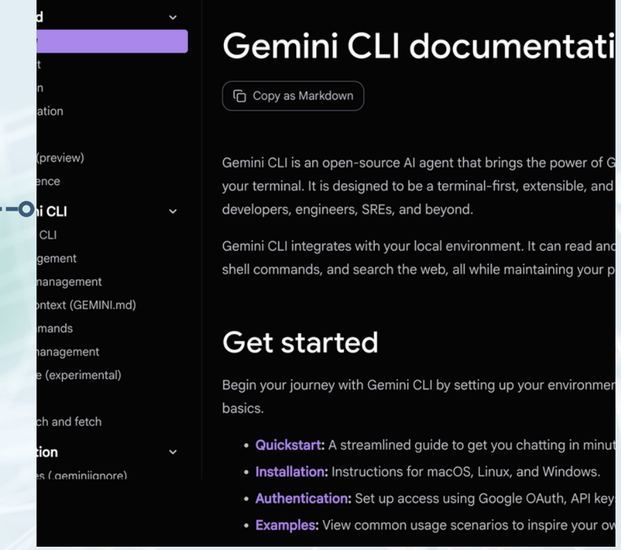


Derivation Process

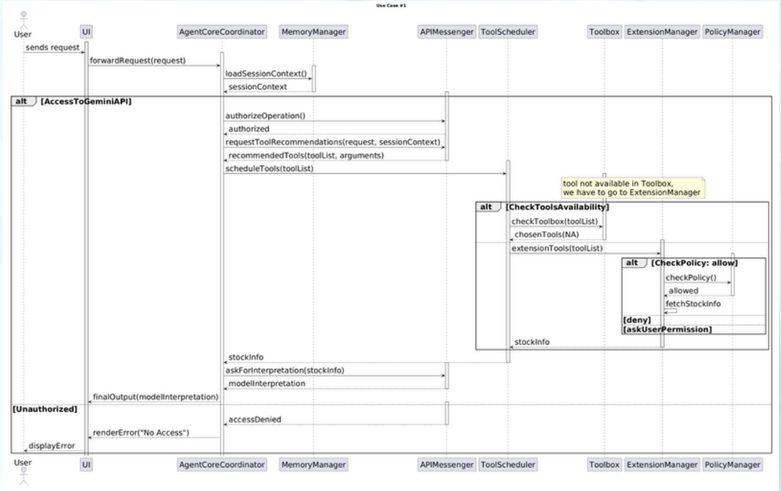
Deciding which Architecture Structure



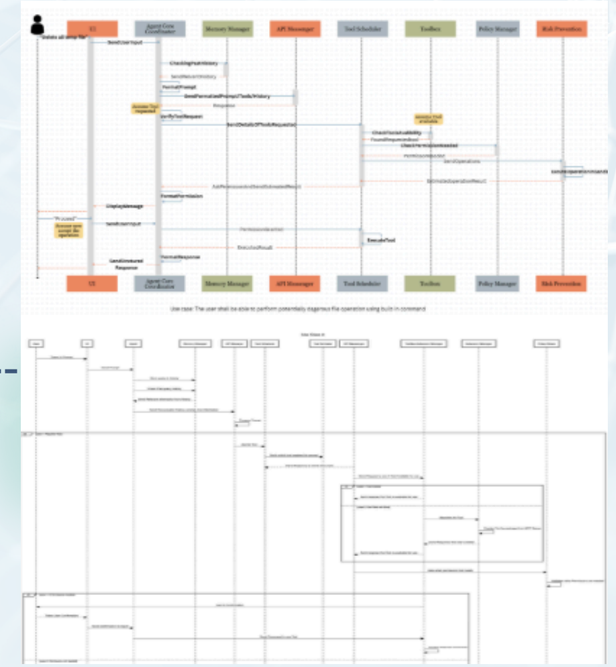
Look at Documentations on Gemini.com



Choosing Final Sequence Diagram



Multiple Different Sequence Diagrams



Architecture Components

01 UI

Serves as a presentation layer, isolates user commands from internal logic

02 Agent-Core Coordinator

Act as a central orchestrator that manages components and interdependencies and execute ReAct loop to transform user request into multi-task step completion

03 API Messenger

Handles Gemini API Authentication, request formatting, error handling, token usage and asynchronous streaming

04 Tool Scheduler

Coordinates Tool Scheduling, validates execution requests and ensure all tools follow policy and schema requirments

05 ToolBox

Used for File manipulation, shell commands, web access, and it creates a sandbox environment for performing operations on user system

06 Policy Manager

Evaluates tool execution requests against a rule set that determines if operations are blocked, permitted or flagged for user confirmation

07 Risk Prevention

Serves as a defensive layer that executes hazardous toolbox operations within isolated environments to prevent damage to host system

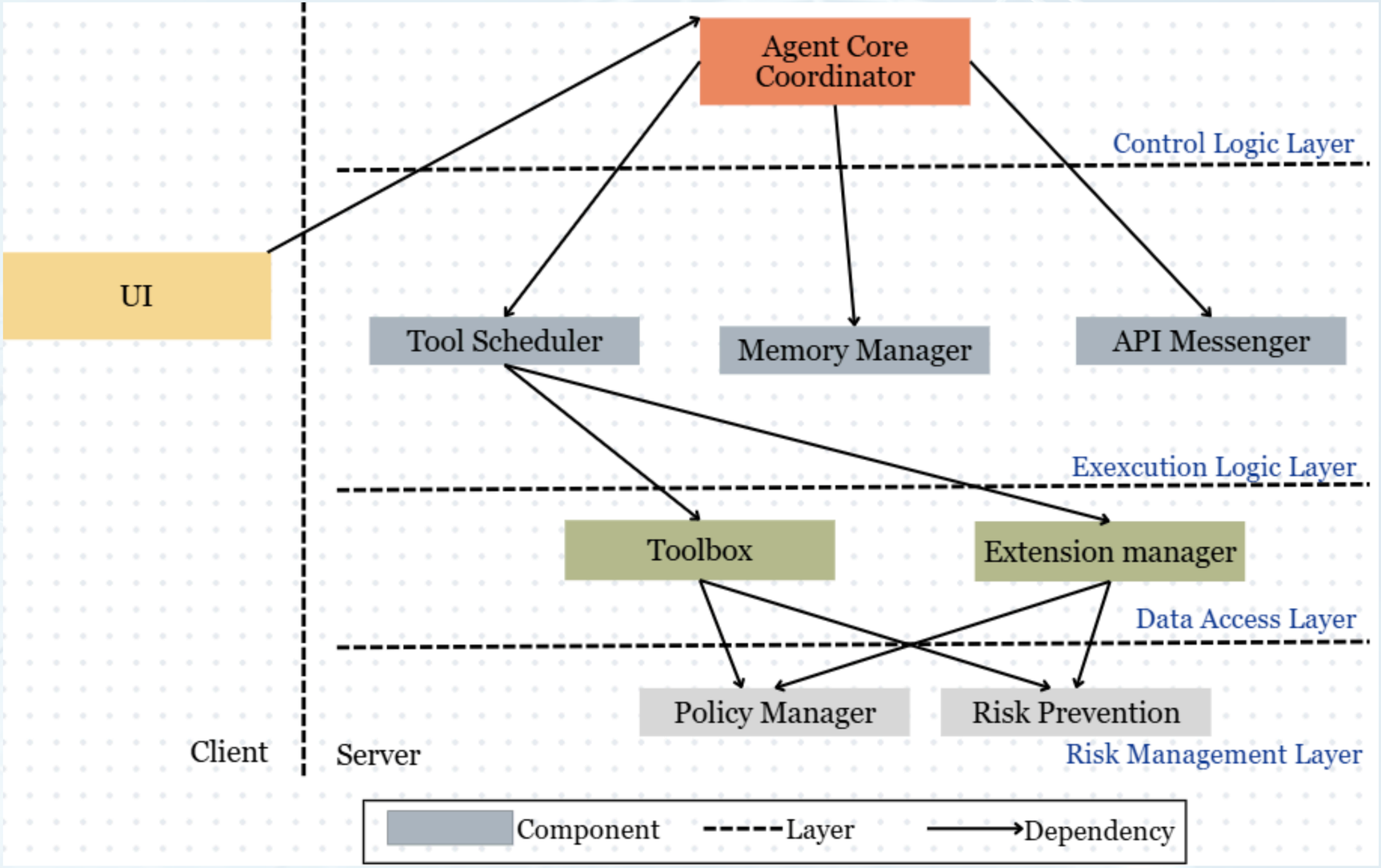
08 Extension Manager

Enables the system to expand its capabilities by connecting and executing external tools, allows modularity

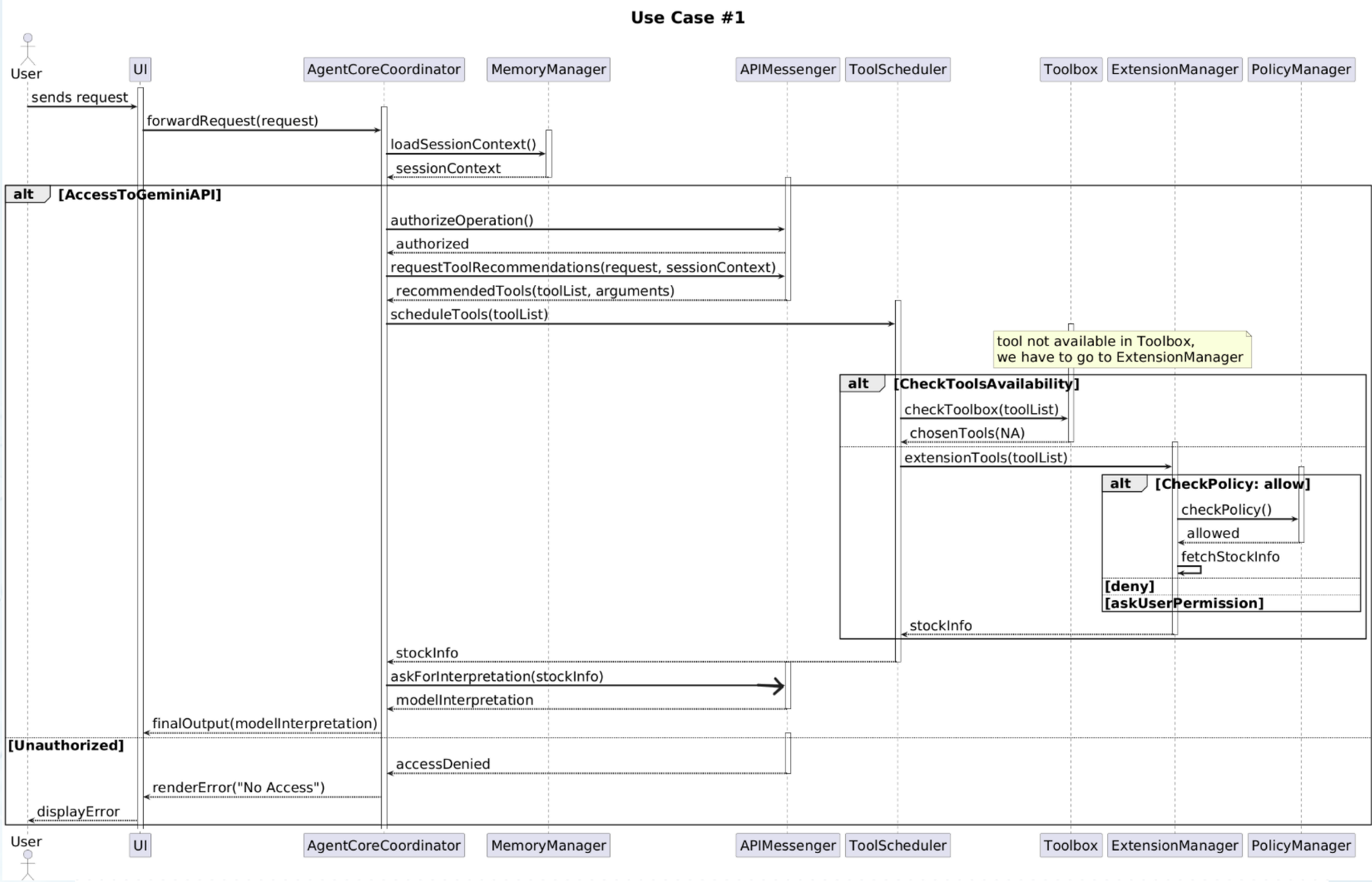
09 Memory Manager

Serves as central data handler for system, responsible for saving conversational history, loading configurations and managaing project files

Box and Line Diagram



Sequence Diagram



Use Case: "User shall be able to ask gemini cli to fetch and summarize stock price."

Architectural Styles

01 Client Server Style

Gemini CLI integrates a Client-Server style where the UI serves as the Client, responsible for capturing input and displaying the final output. The Control, Execution, Data Process, and Risk Management layers collectively form the Server, which receives user requests to utilize AI reasoning, API communication, and tool operations before sending results back to the client.

02 Layered Style

Gemini CLI Layered Architecture builds on a multi-layer foundation where the control logic manages the orchestration, and the execution logic performs specific tasks. With this structure it allows modularity, as each layer can evolve separately while maintaining stability and separation of concerns

Alternative Architectural Style

03

Publish and Subscribe Style

Architectural Approach: Agent Core Coordinator acts as a central broker to delegate tasks and ensure extensibility.

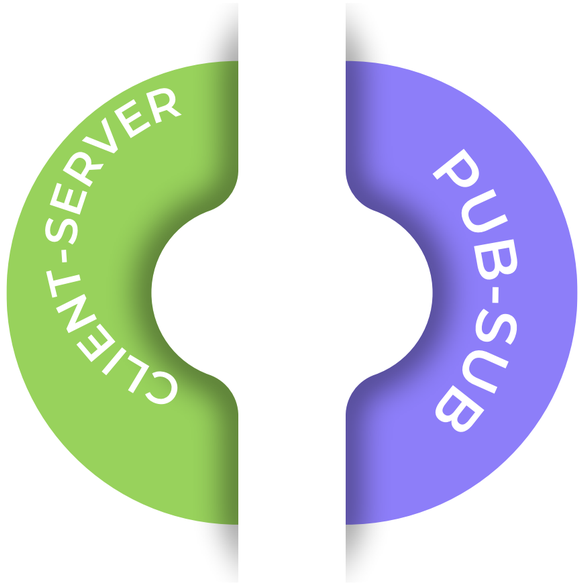
Modularity: Developers can add new capabilities or swap models without modifying core codebase

Trade Off: High degree decoupling results in more potential risk for unpredictable behaviours.

Limitations of each Architecture Style

Performance Latency

Sinkhole Effect



Lack of Control

Complex Debugging

Lessons Learnt

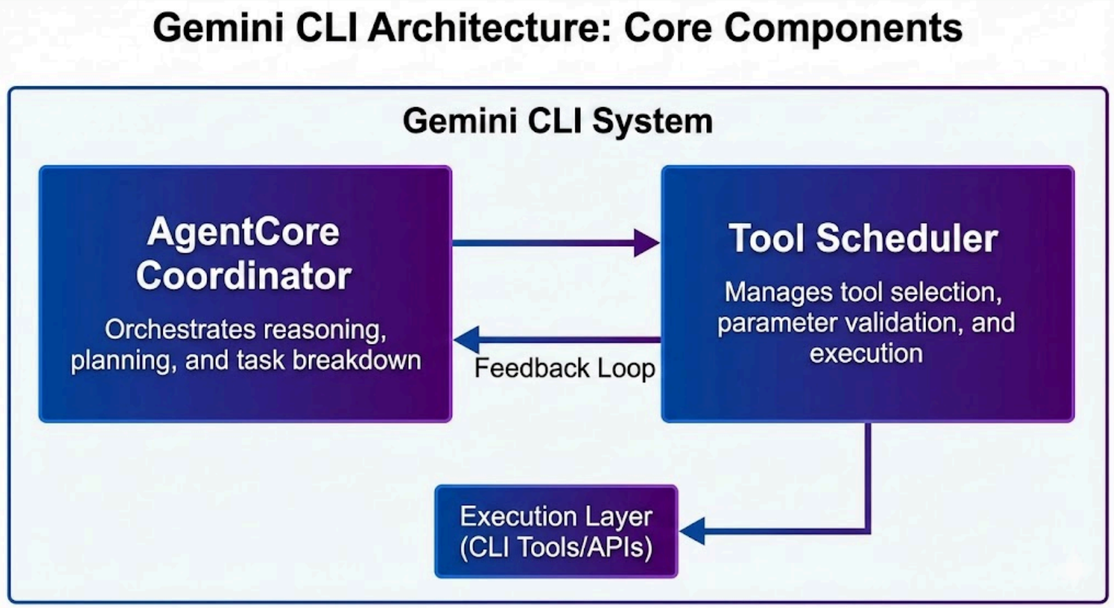
Lesson 1:

Conceptual Architecture vs Code Structure

Lesson 2

Dividing work does not mean isolation

Understanding boundaries between components!



Limitations

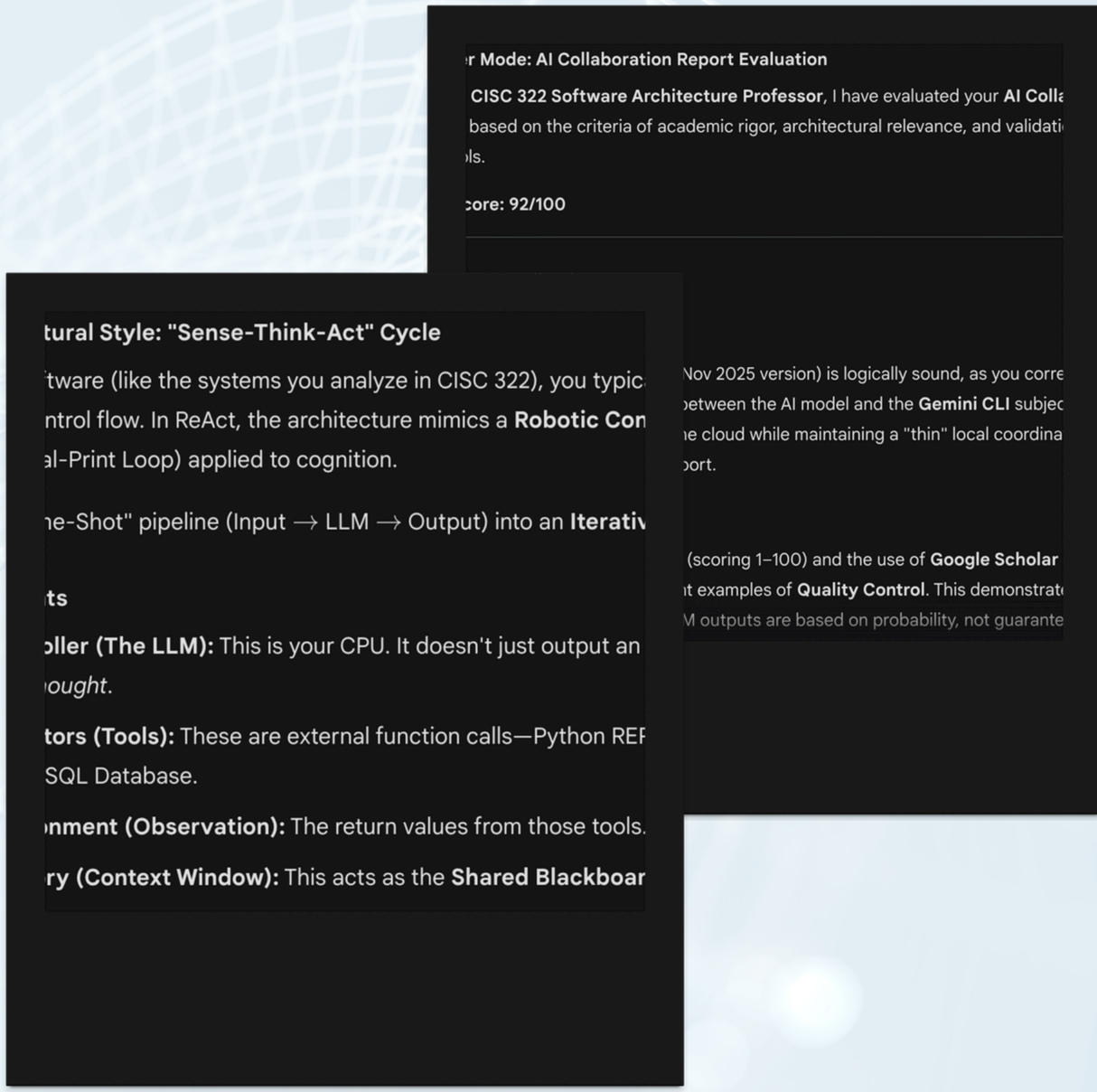
USAGE 1

Understanding new terminology

USAGE 2

Academic Quality Control

AI Teammate




THANK YOU