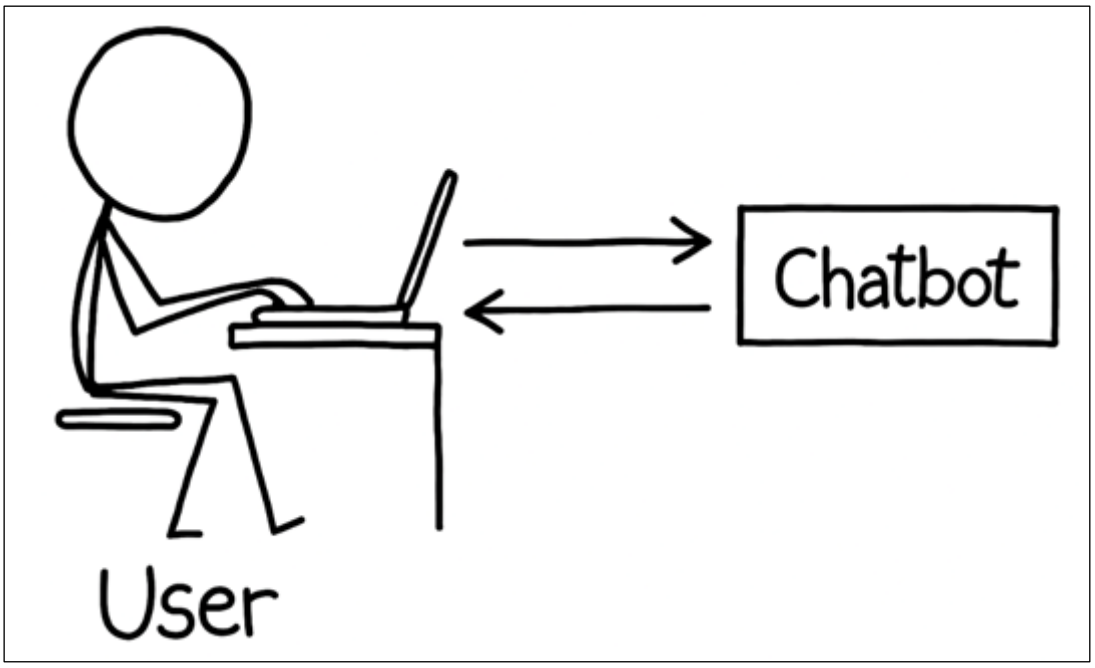
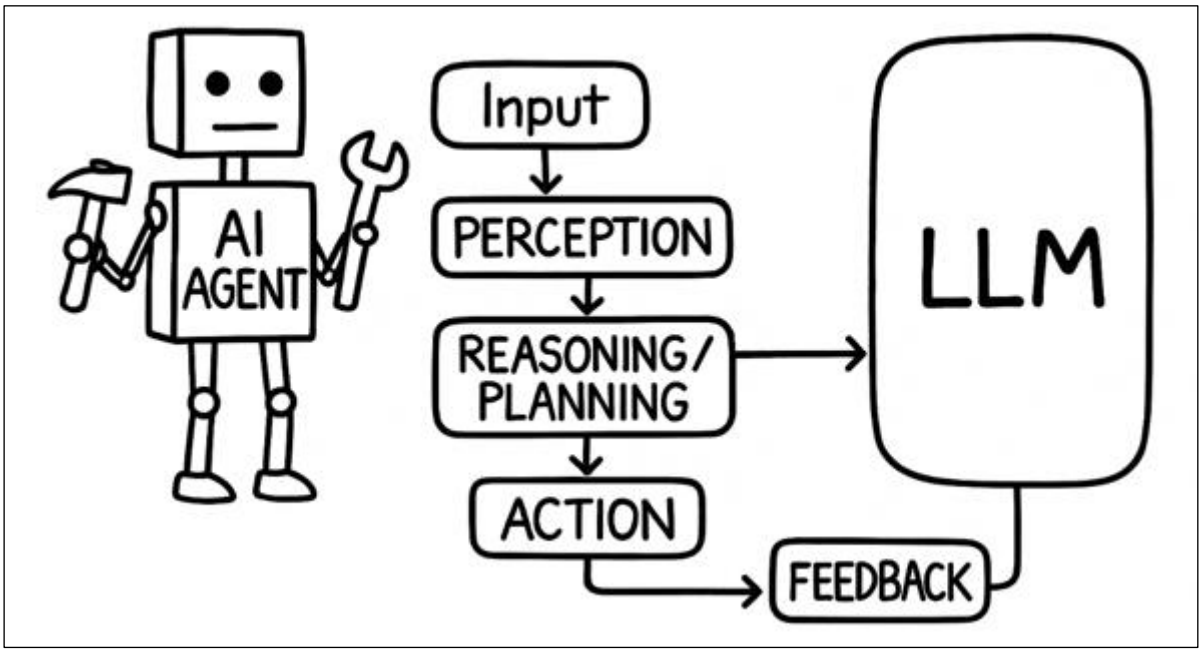
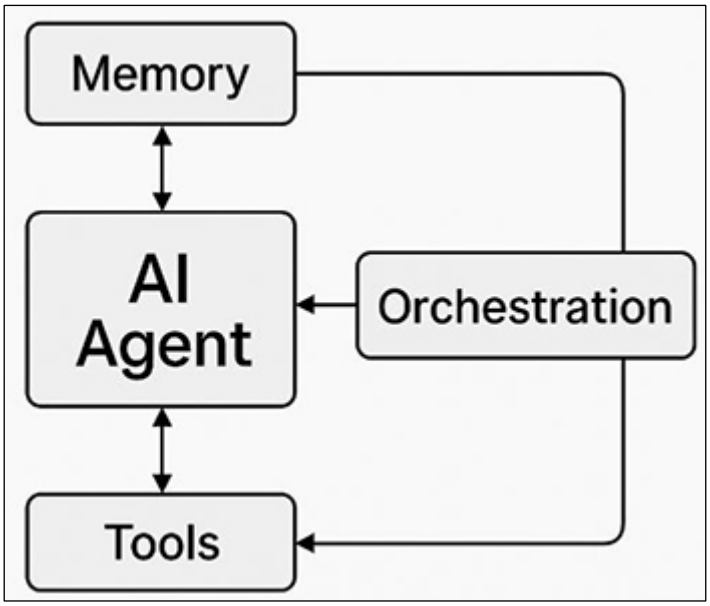
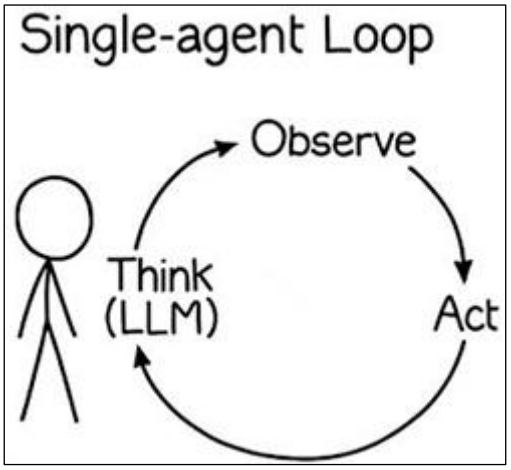
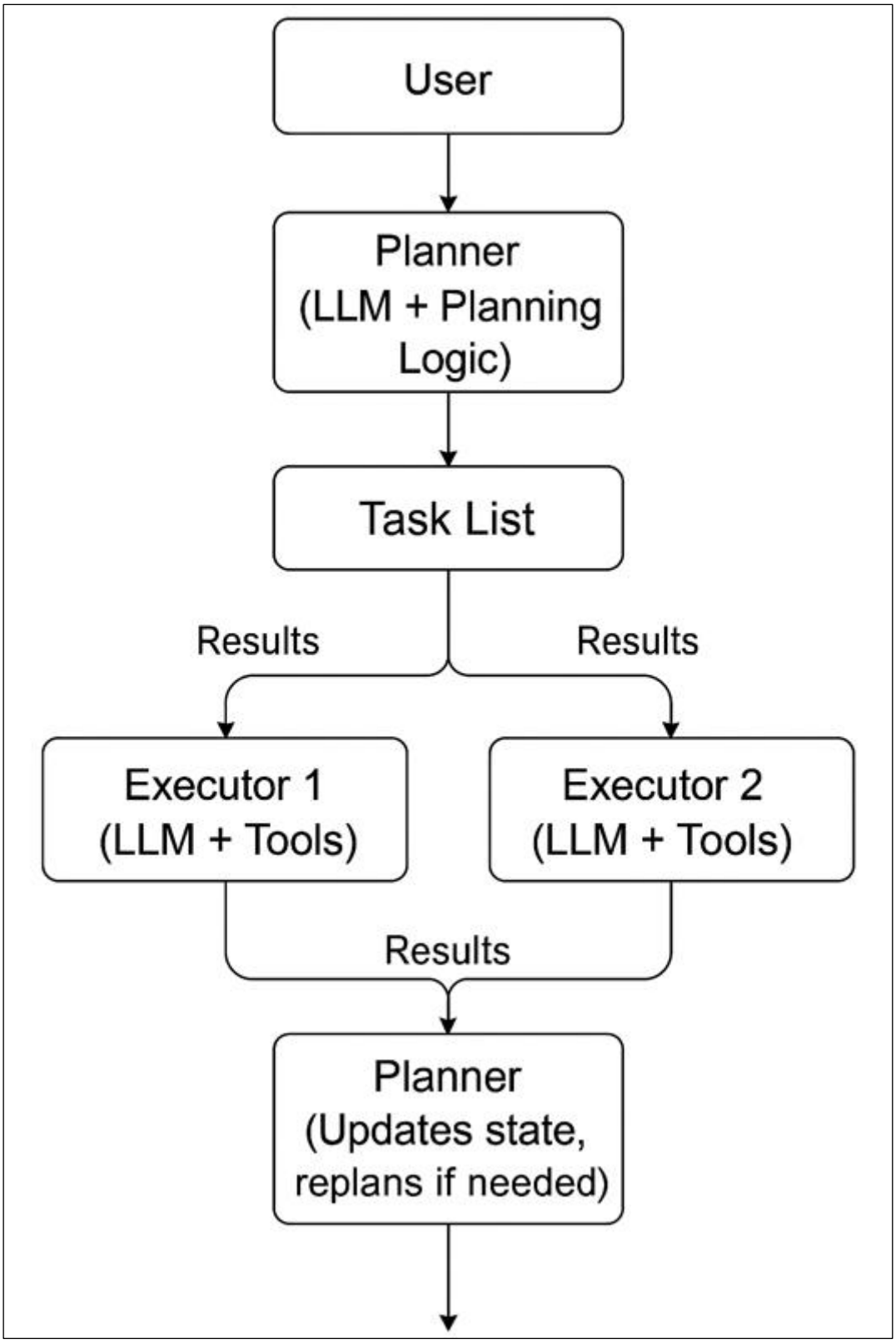
Chapter 1: Introduction to Generative AI and AI Agents

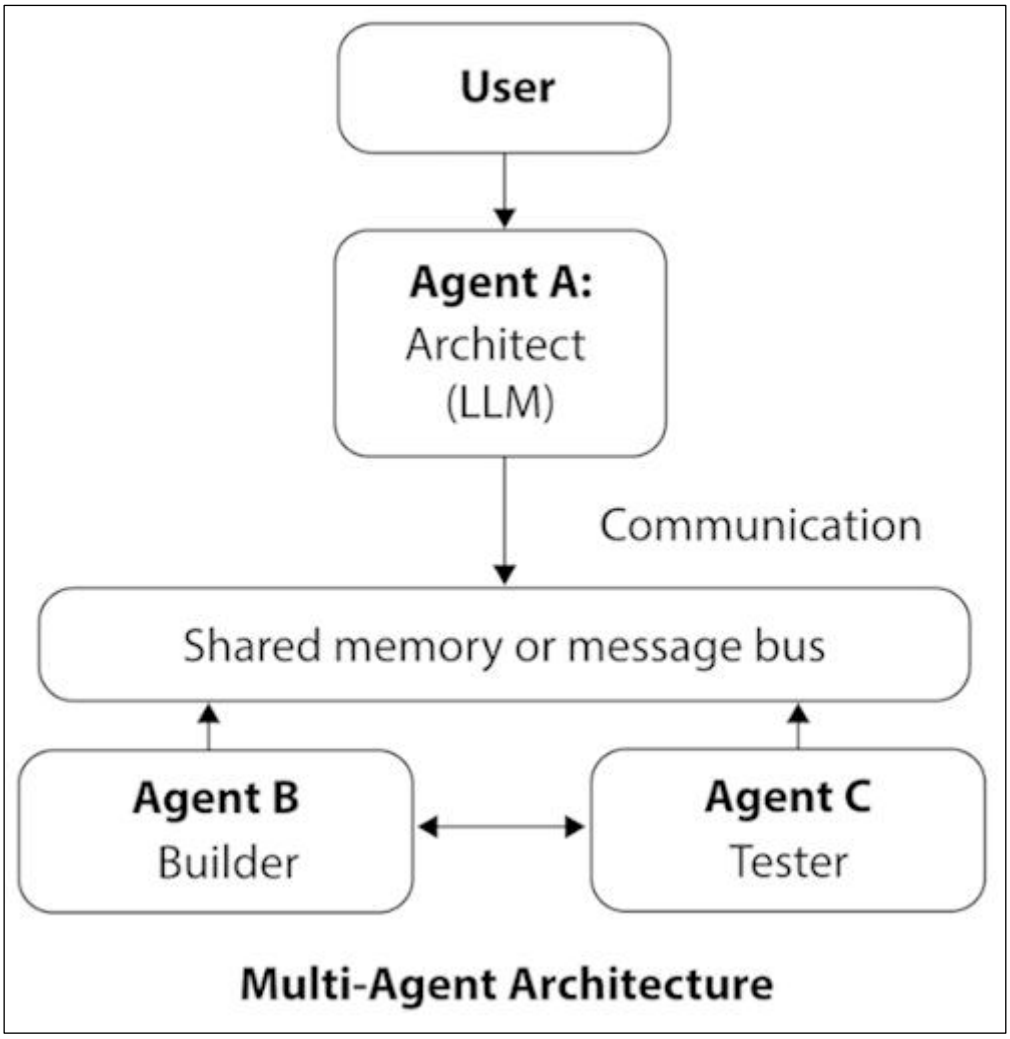





Memory
AI Agent
Orchestration
Tools

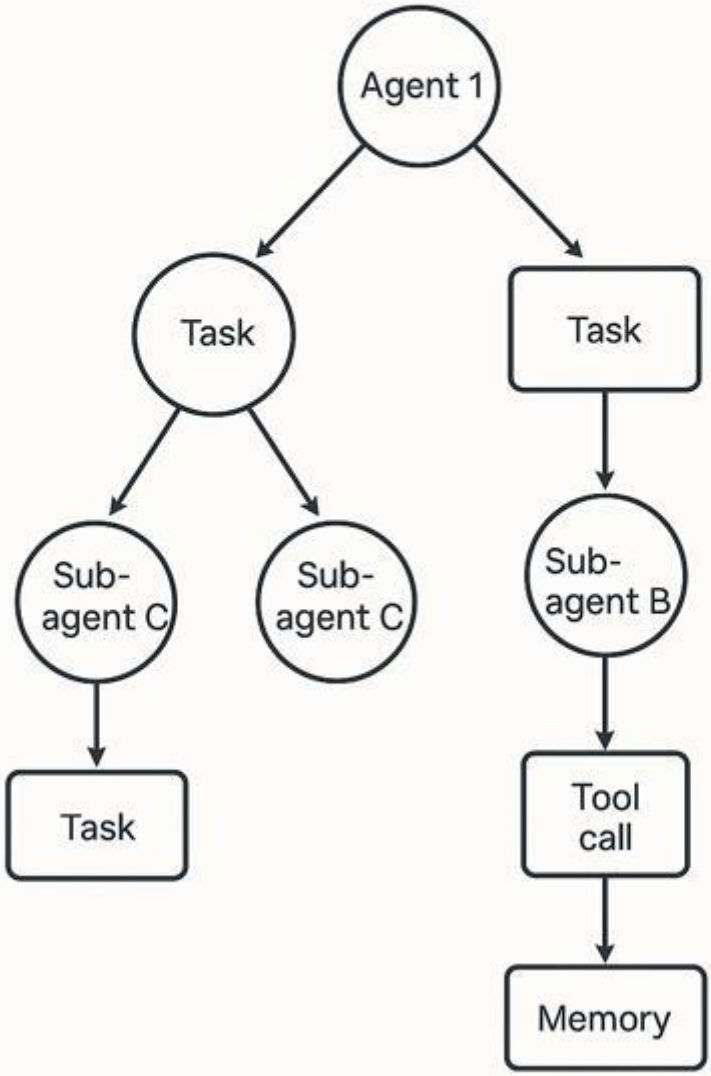

Single-agent Loop
Observe
Think (LLM)
Act


User
Planner
(LLM + Planning
Logic)
Task List
Results
Results
Executor 1
(LLM + Tools)
Executor 2
(LLM + Tools)
Results
Planner
(Updates state,
replans if needed)

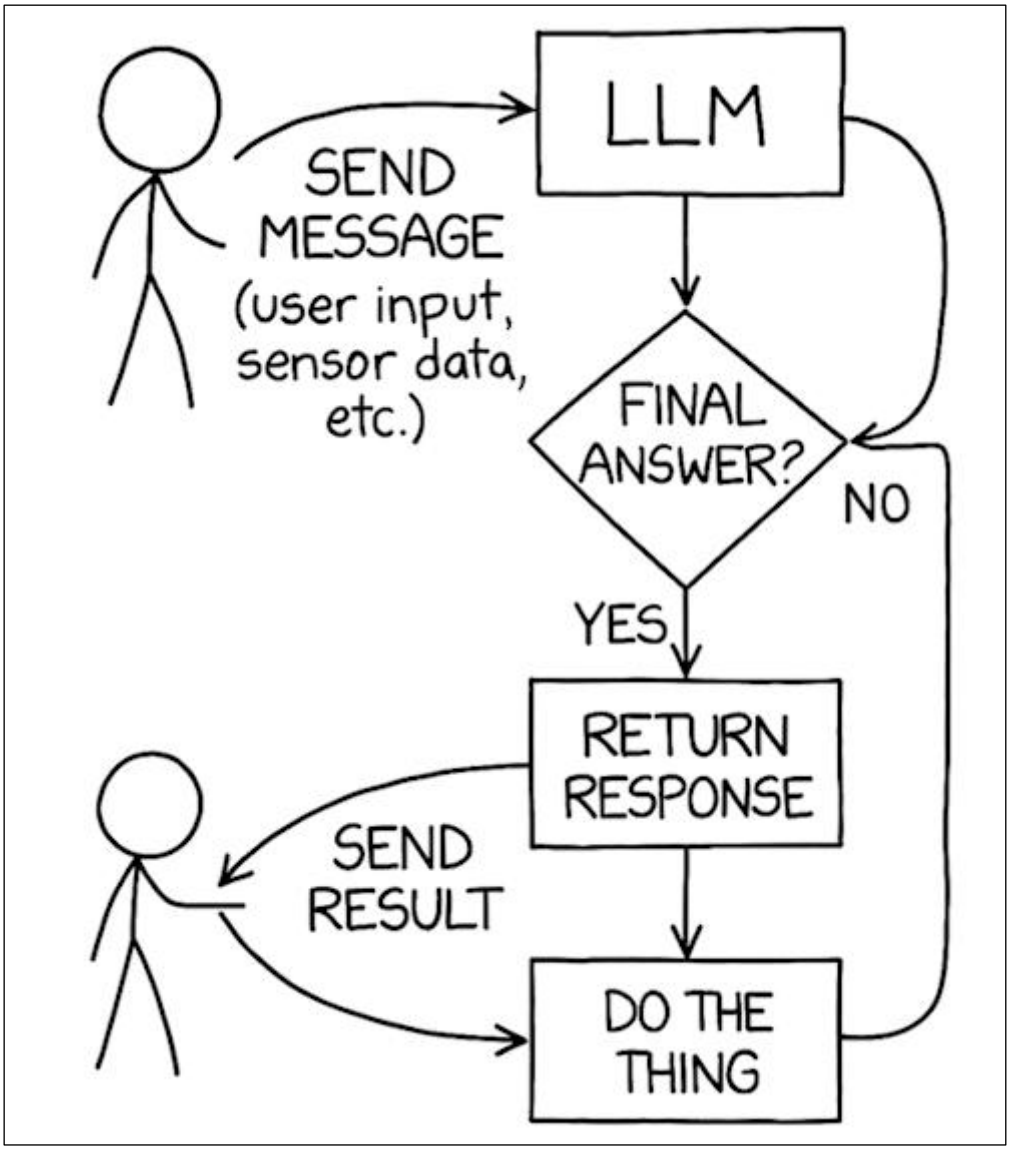


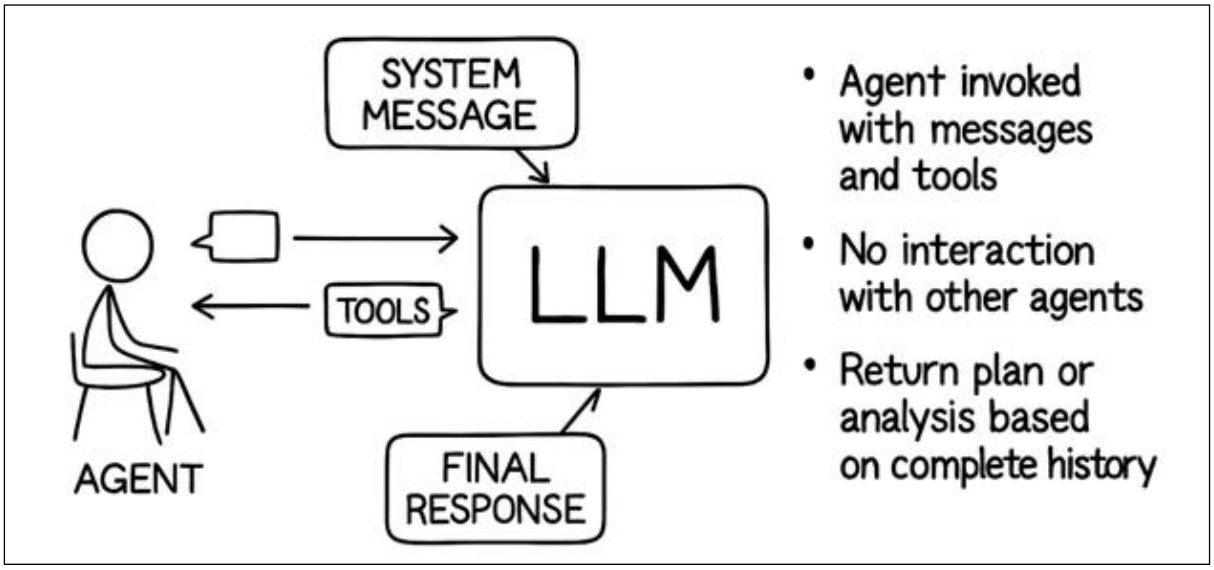
Multi-Agent Architecture

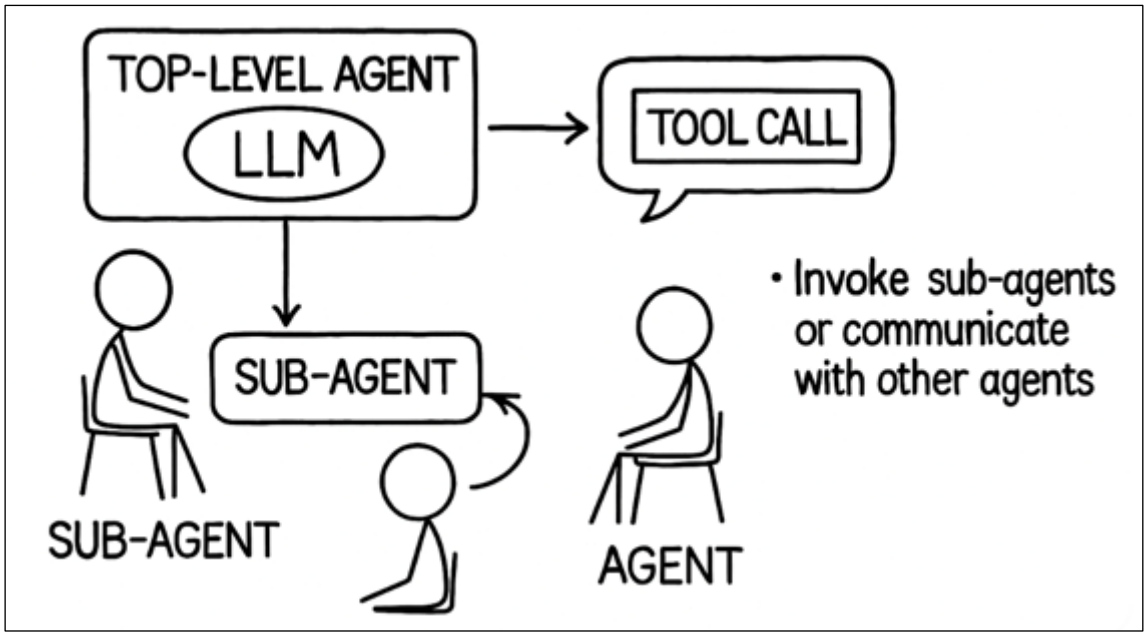
Graph-based architectures represent the relationships between agents, tasks, tool invocations, and memory updates as a graph.

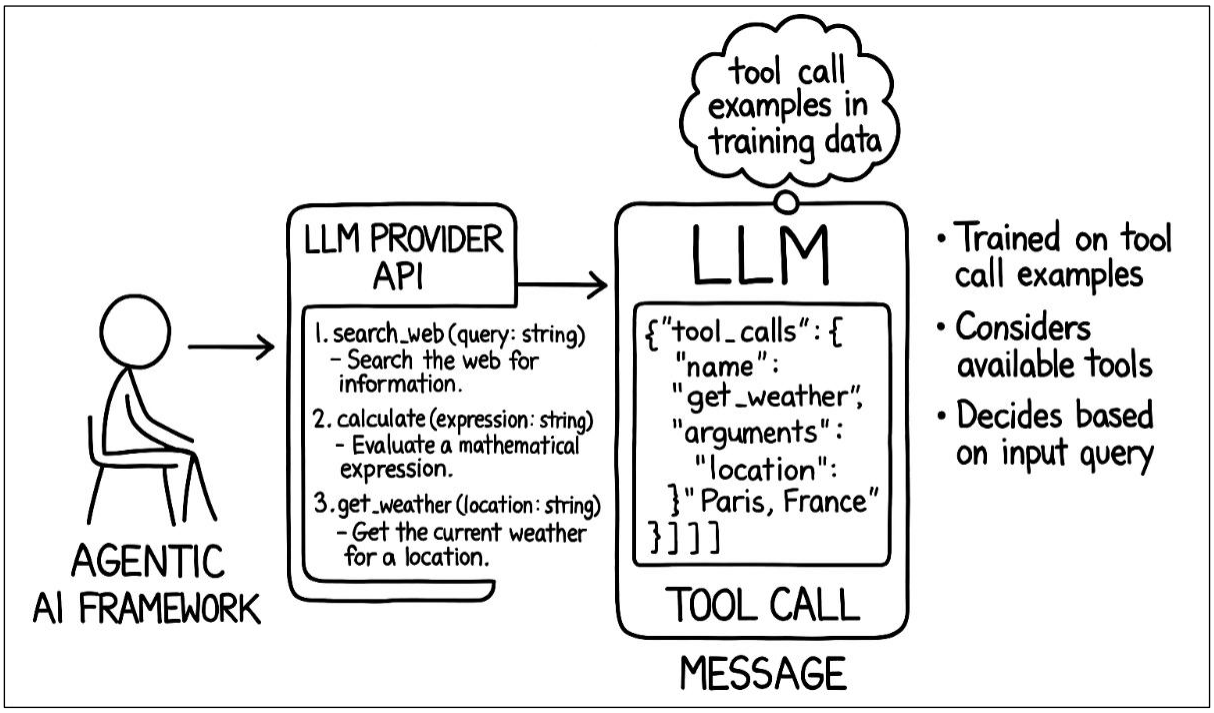


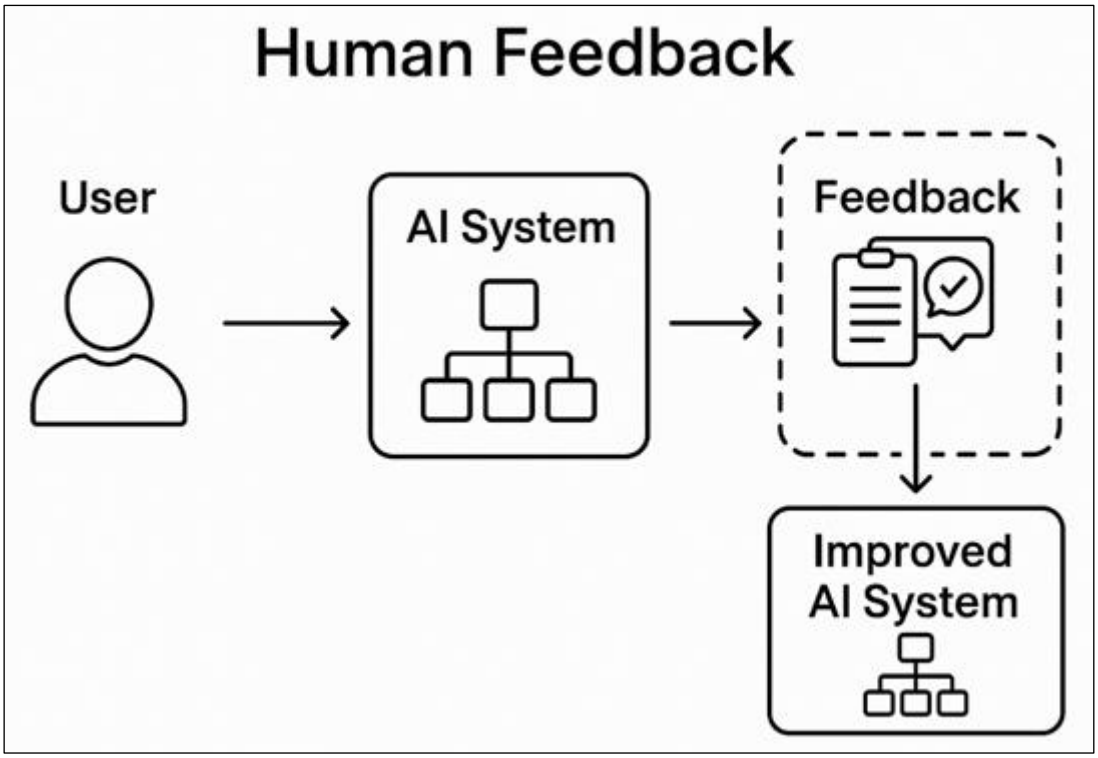
Chapter 2: Understanding How AI Agents Work

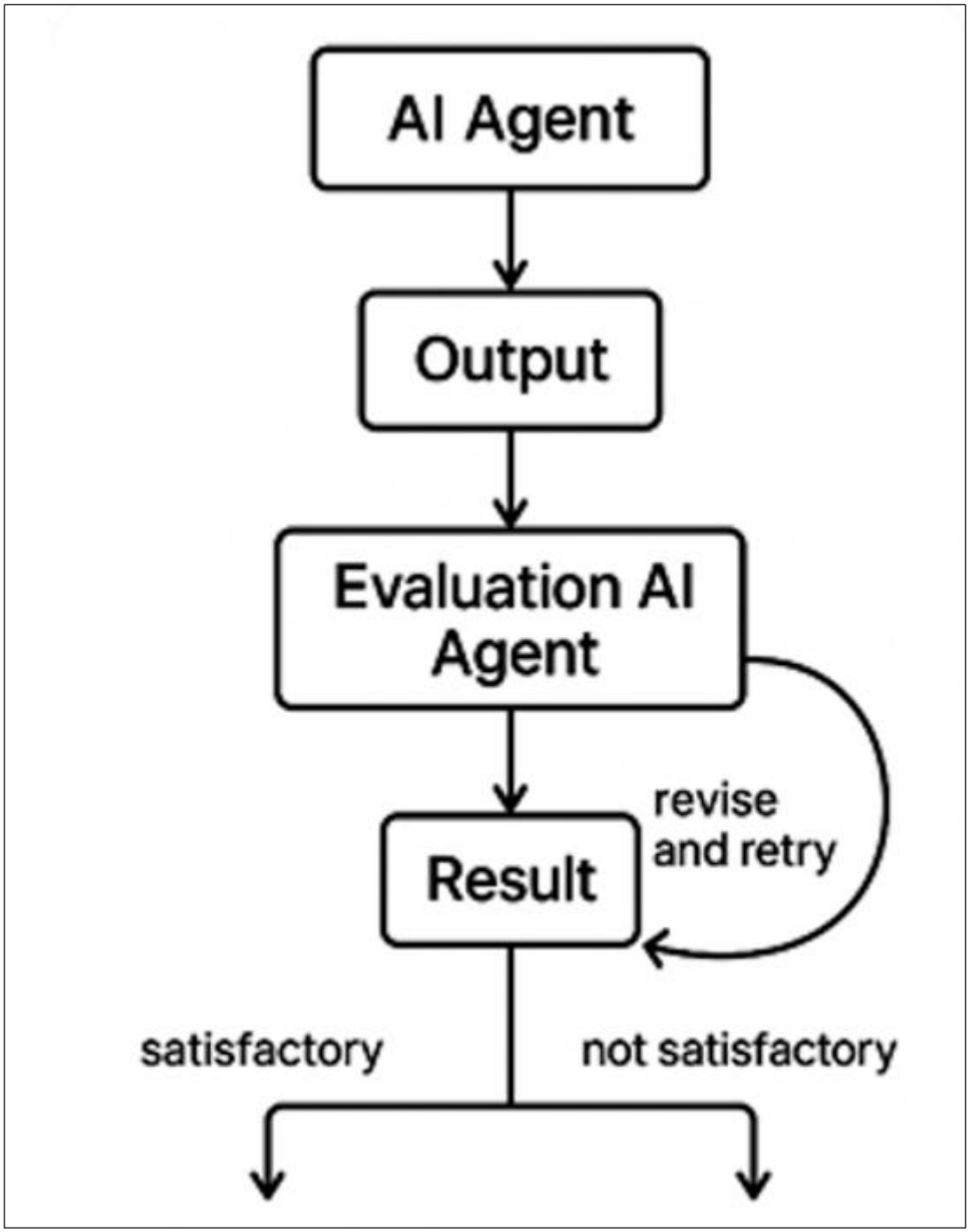


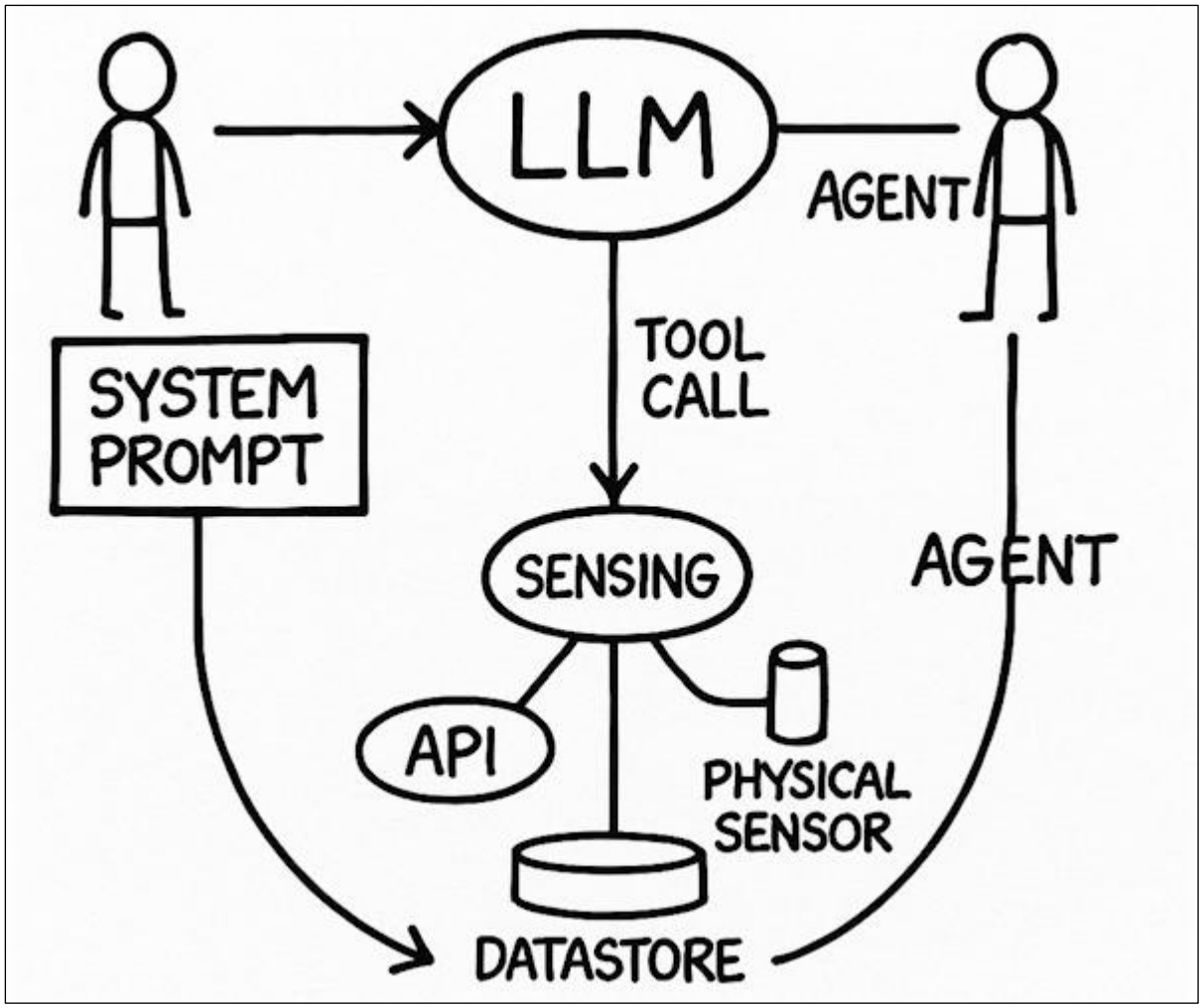


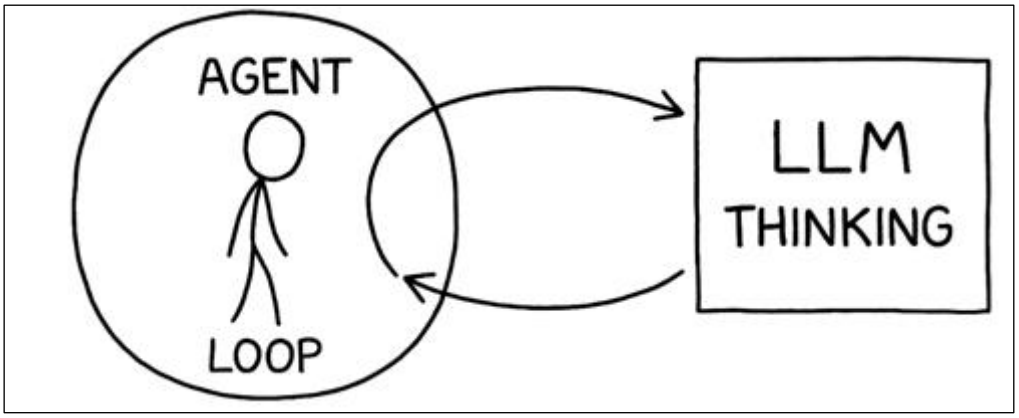

TOP-LEVEL AGENT
LLM
TOOL CALL
SUB-AGENT
SUB-AGENT
AGENT
• Invoke sub-agents or communicate with other agents

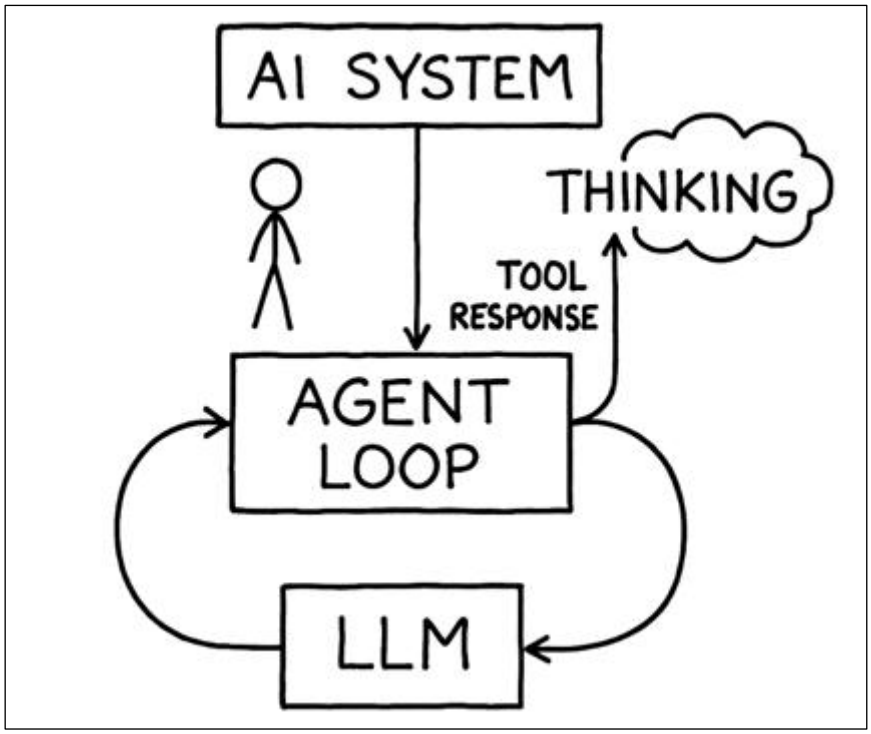

tool call examples in training data
LLM PROVIDER API
1. search_web(query: string) - Search the web for information.
2. calculate(expression: string) - Evaluate a mathematical expression.
3. get_weather(location: string) - Get the current weather for a location.
LLM
{"tool_calls": [{ "name": "get_weather", "arguments": { "location": "Paris, France" }}]}
TOOL CALL
MESSAGE
AGENTIC AI FRAMEWORK
• Trained on tool call examples
• Considers available tools
• Decides based on input query

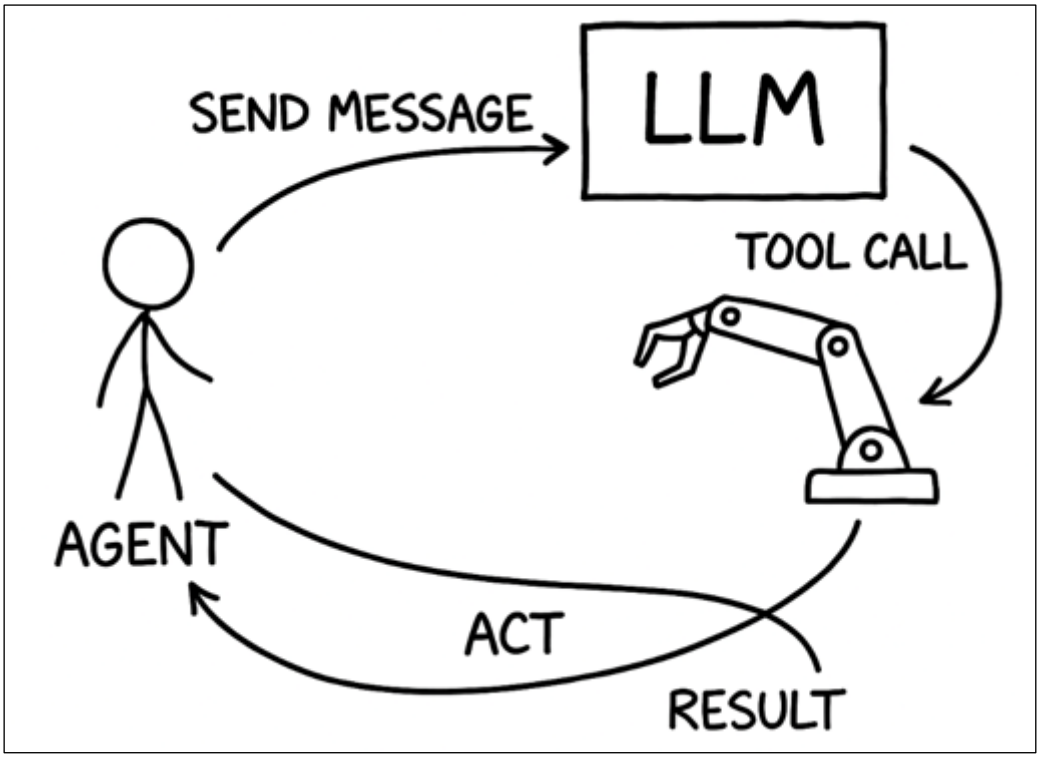

Human Feedback
User
AI System
Feedback
Improved
AI System

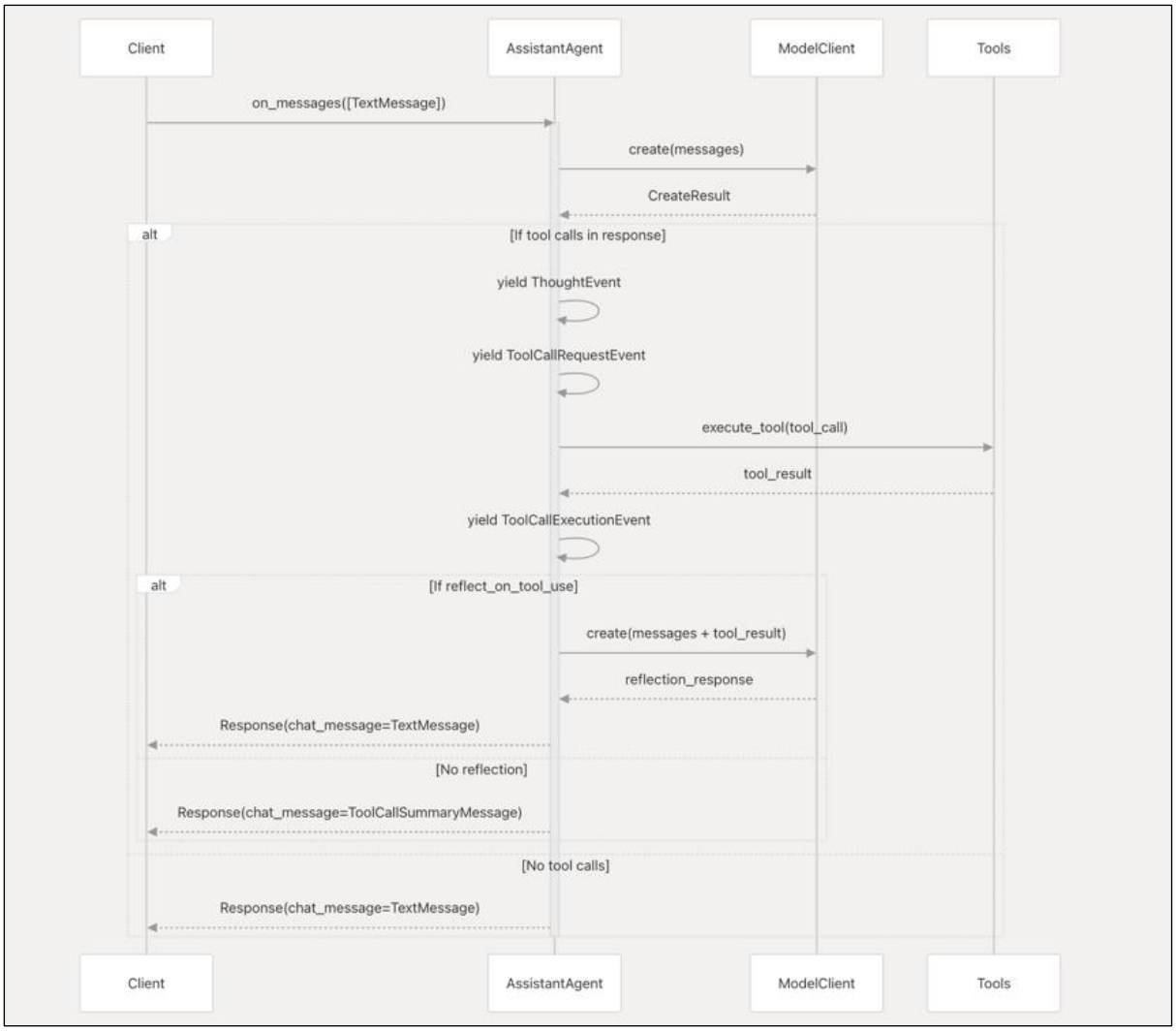

AI Agent
Output
Evaluation AI Agent
revise and retry
Result
satisfactory
not satisfactory

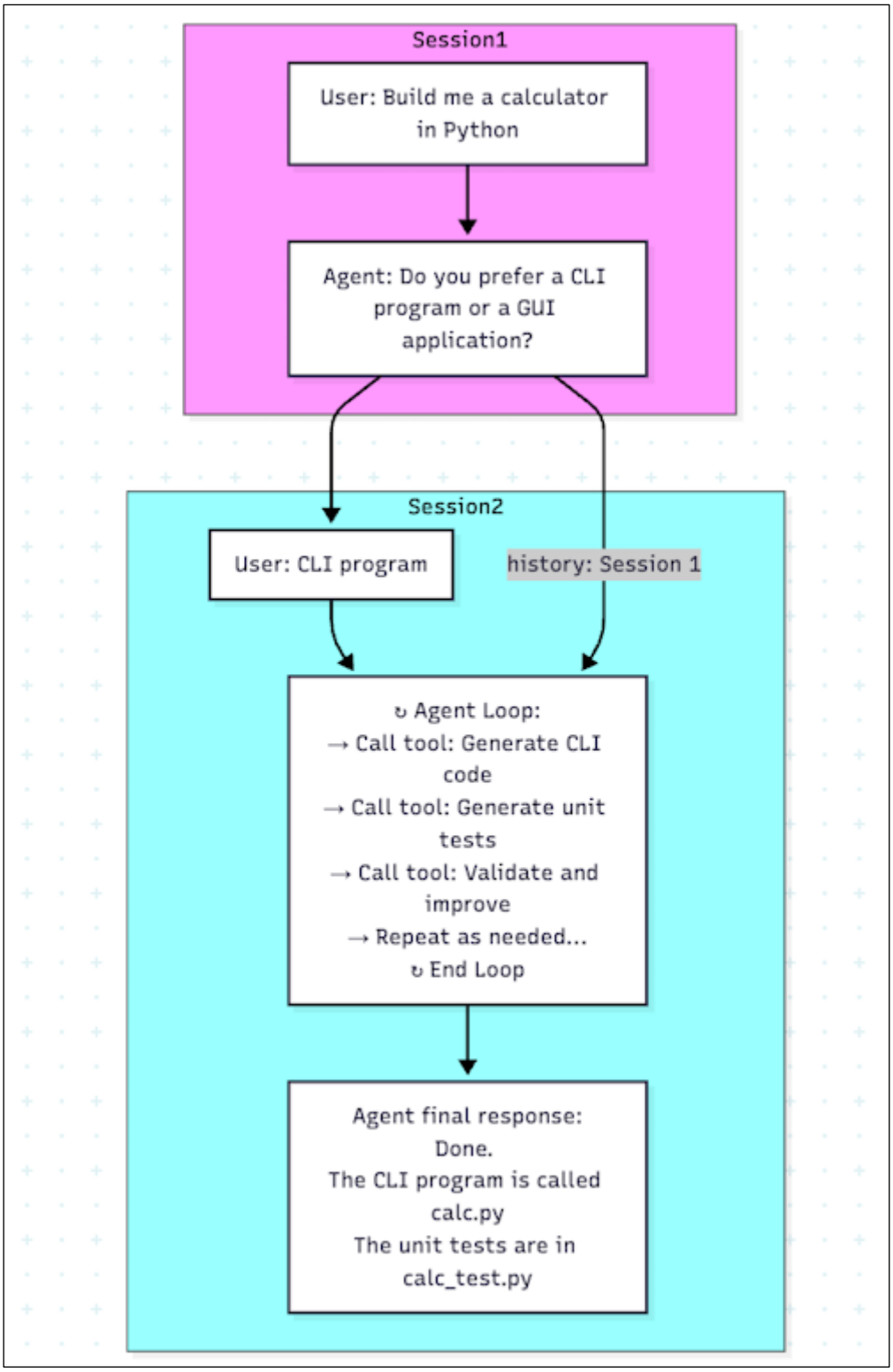

LLM
AGENT
TOOL
CALL
SYSTEM
PROMPT
SENSING
AGENT
API
PHYSICAL
SENSOR
DATASTORE

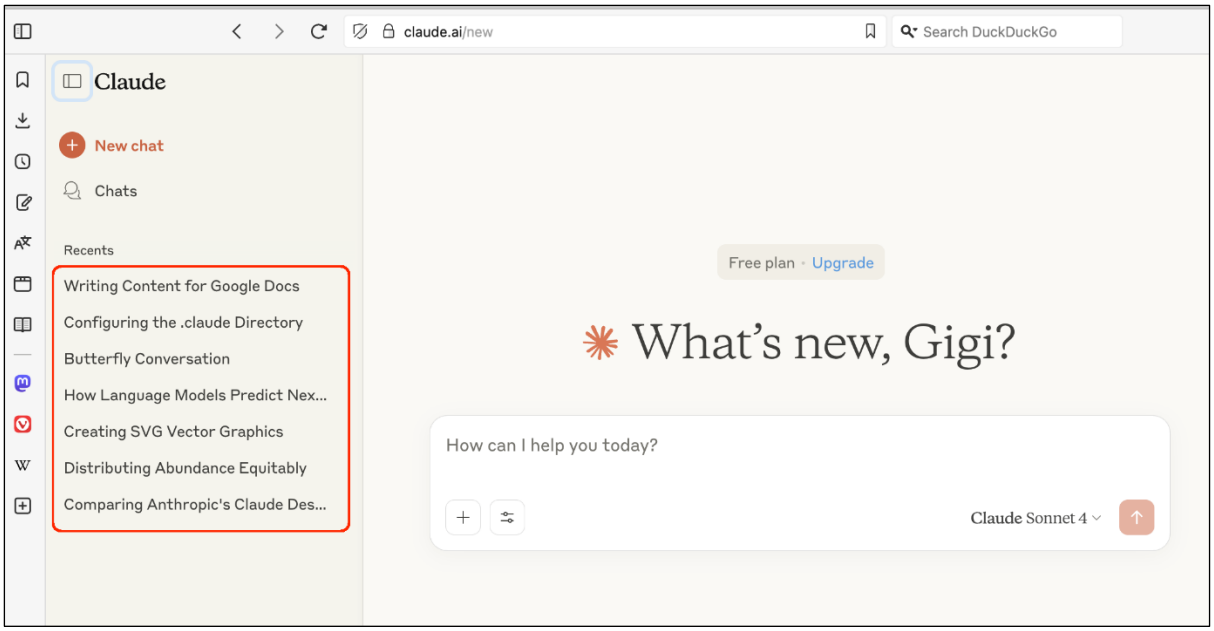

AGENT
LOOP
LLM
THINKING

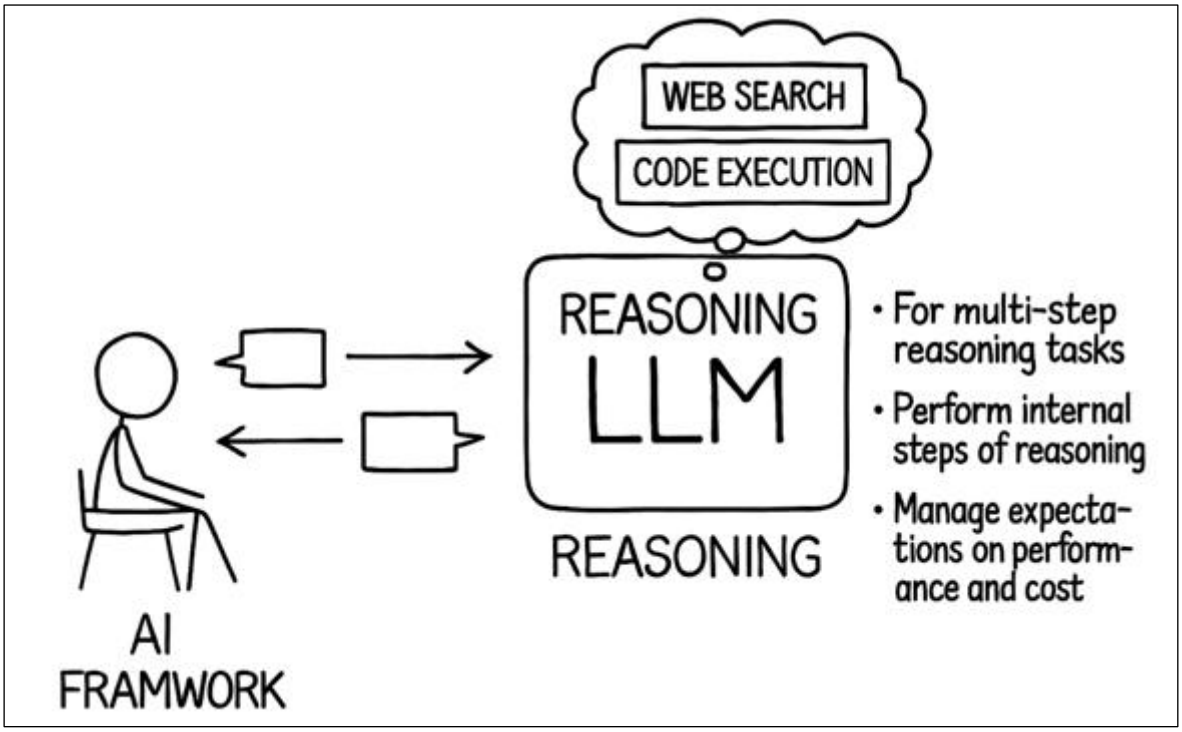

AI SYSTEM
THINKING
TOOL
RESPONSE
AGENT
LOOP
LLM


SEND MESSAGE
LLM
TOOL CALL
AGENT
ACT
RESULT

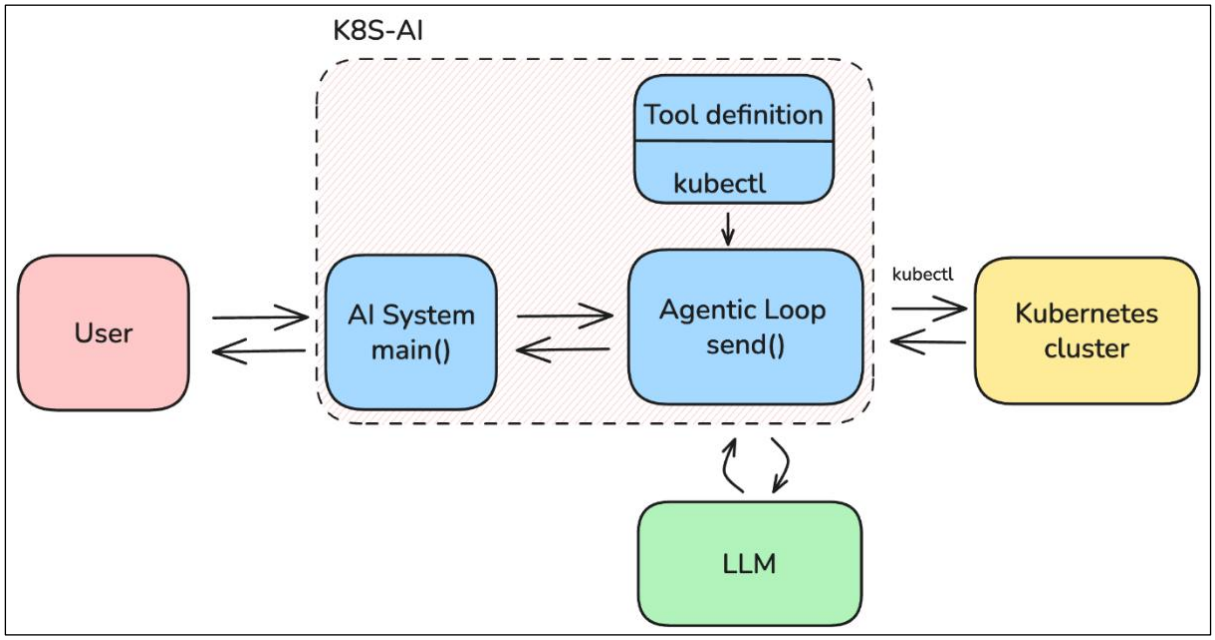

Client
AssistantAgent
ModelClient
Tools
on_messages([TextMessage])
create(messages)
CreateResult
alt
[If tool calls in response]
yield ThoughtEvent
yield ToolCallRequestEvent
execute_tool(tool_call)
tool_result
yield ToolCallExecutionEvent
alt
[If reflect_on_tool_use]
create(messages + tool_result)
reflection_response
Response(chat_message=TextMessage)
[No reflection]
Response(chat_message=ToolCallSummaryMessage)
[No tool calls]
Response(chat_message=TextMessage)
Client
AssistantAgent
ModelClient
Tools


Session1
User: Build me a calculator in Python
Agent: Do you prefer a CLI program or a GUI application?
Session2
User: CLI program
history: Session 1
↻ Agent Loop:
→ Call tool: Generate CLI code
→ Call tool: Generate unit tests
→ Call tool: Validate and improve
→ Repeat as needed...
↻ End Loop
Agent final response:
Done.
The CLI program is called calc.py
The unit tests are in calc_test.py

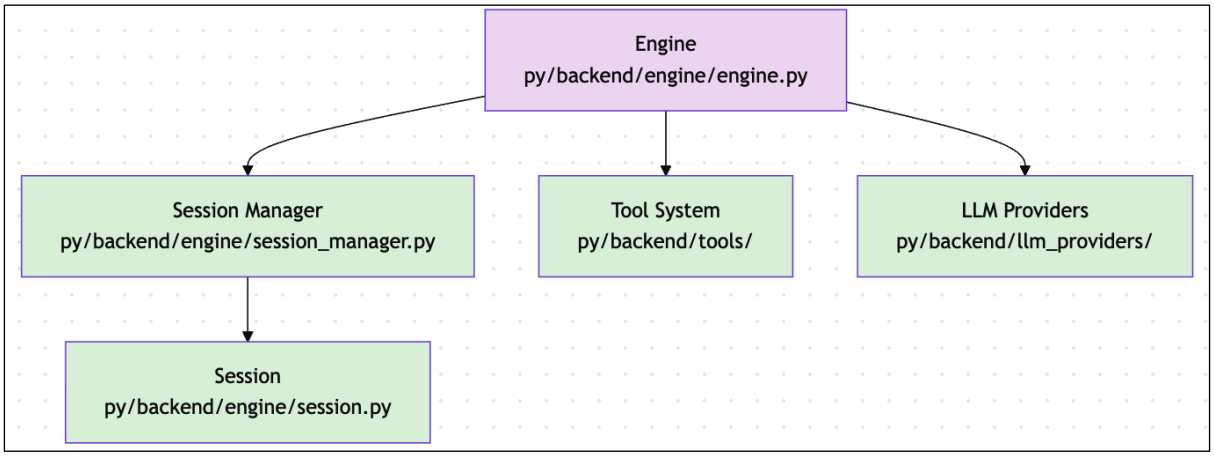

claude.ai/new
Search DuckDuckGo
Claude
New chat
Chats
Recents
Writing Content for Google Docs
Configuring the .claude Directory
Butterfly Conversation
How Language Models Predict Nex...
Creating SVG Vector Graphics
Distributing Abundance Equitably
Comparing Anthropic's Claude Des...
Free plan · Upgrade
What's new, Gigi?
How can I help you today?
Claude Sonnet 4

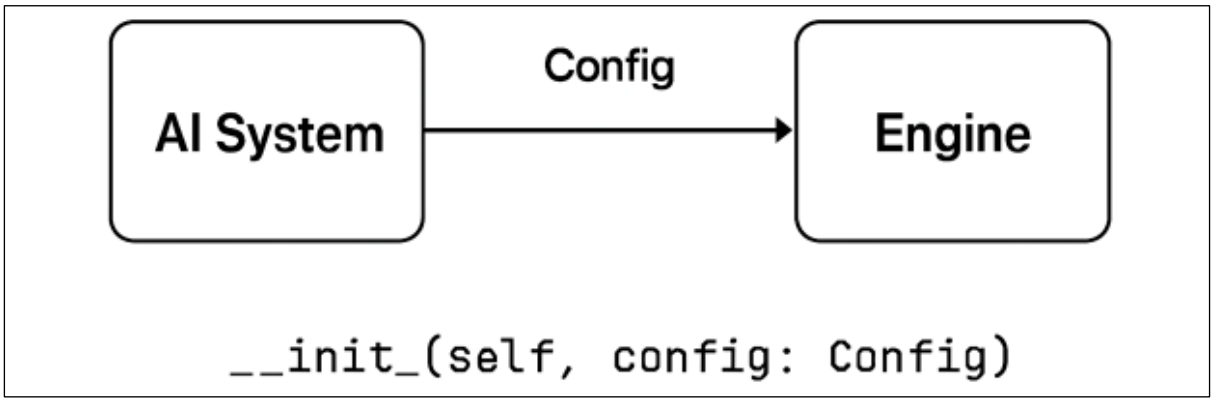

WEB SEARCH
CODE EXECUTION
REASONING
LLM
REASONING
AI
FRAMWORK
• For multi-step reasoning tasks
• Perform internal steps of reasoning
• Manage expecta-tions on perform-ance and cost

Chapter 3: A Hands-On Walk-Through of a Simple AI Agent

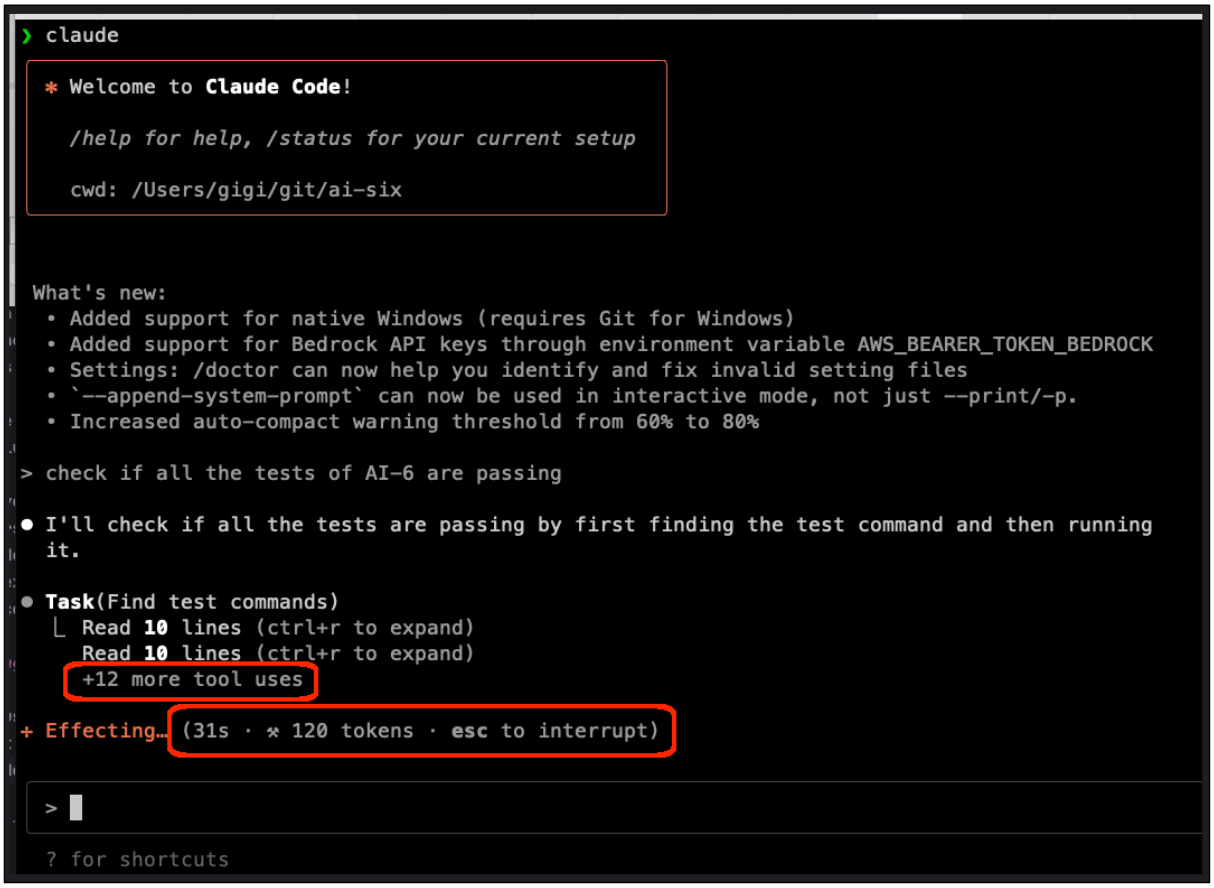


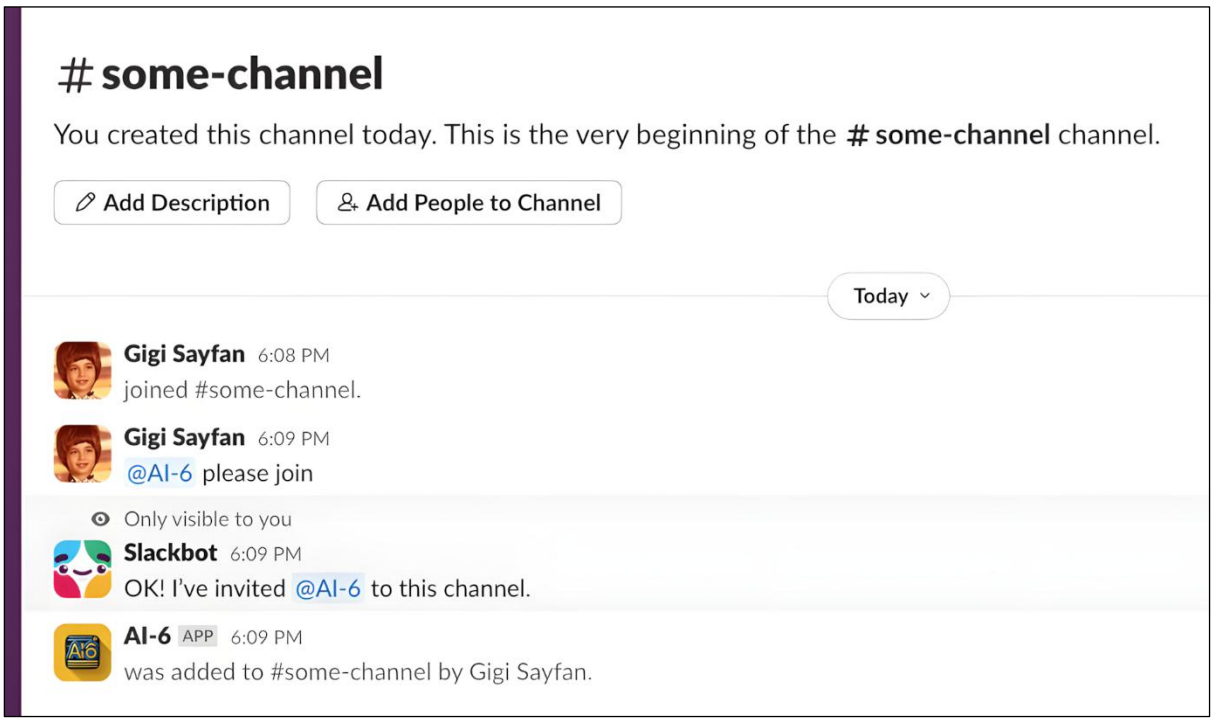
Chapter 4: Building a Tool-Based Agentic AI Framework

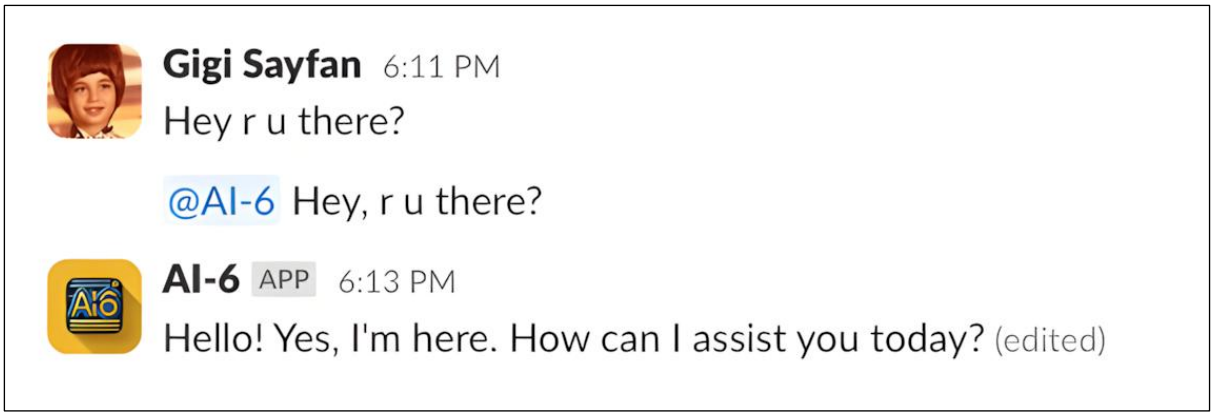


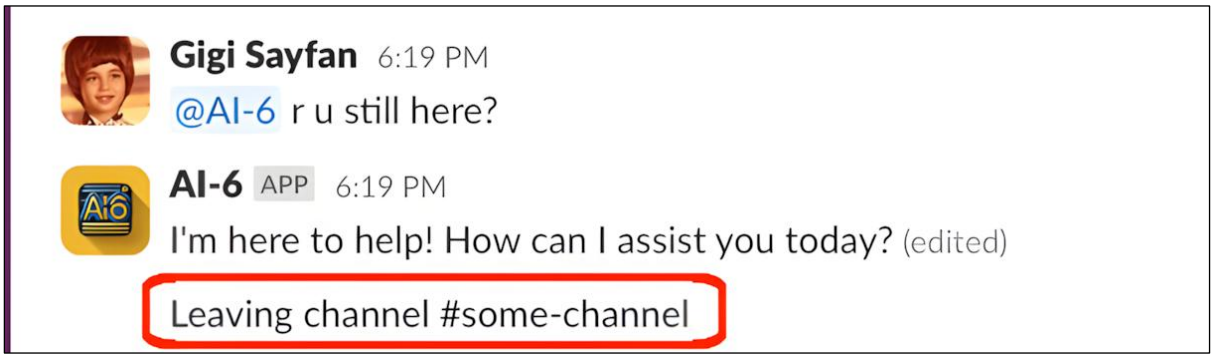


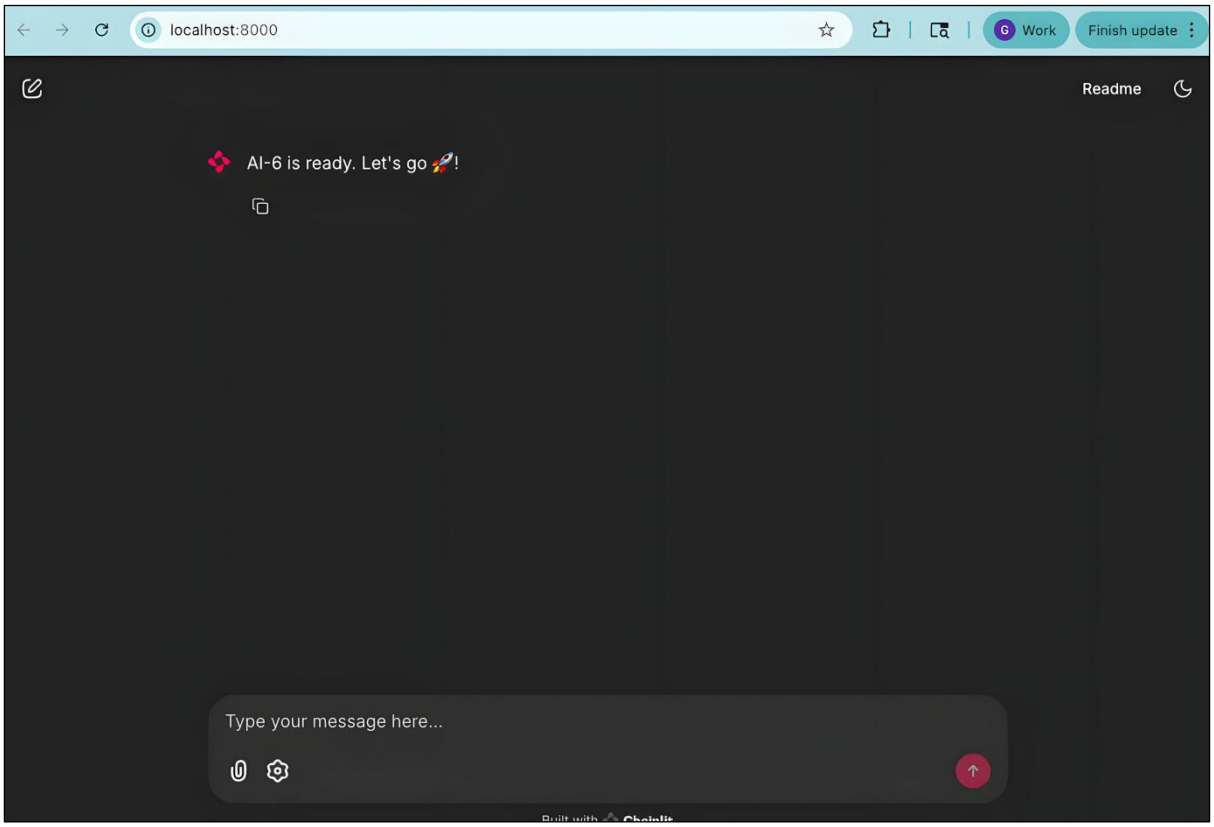
Chapter 6: Creating Chat Interfaces Using Slack and C

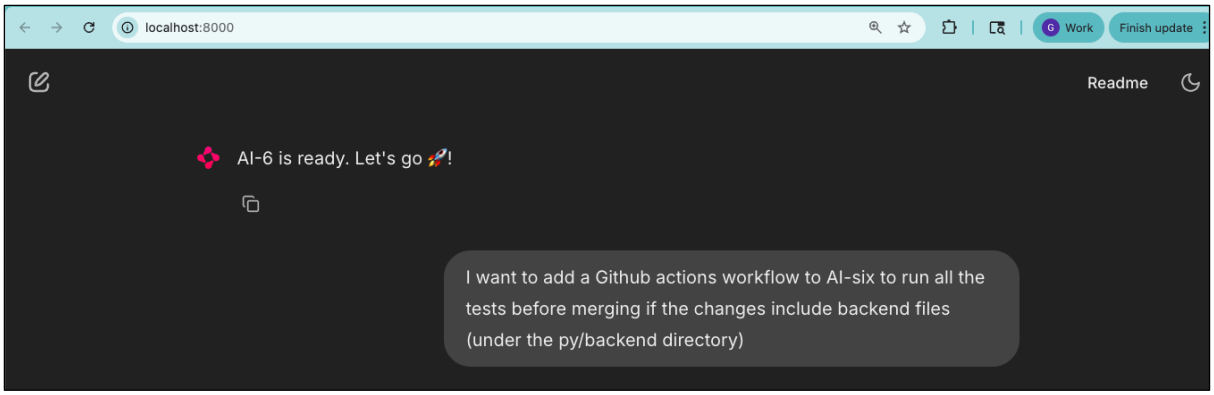





Gigi Sayfan 6:11 PM
Hey r u there?
@AI-6 Hey, r u there?
AI-6 APP 6:13 PM
Hello! Yes, I'm here. How can I assist you today? (edited)


Gigi Sayfan 6:19 PM
@AI-6 r u still here?
AI-6 APP 6:19 PM
I'm here to help! How can I assist you today? (edited)
Leaving channel #some-channel


localhost:8000
Work
Finish update
Readme
AI-6 is ready. Let's go 🚀!
Type your message here...


localhost:8000
Work
Finish update
Readme
AI-6 is ready. Let's go 🚀!
I want to add a Github actions workflow to AI-six to run all the tests before merging if the changes include backend files (under the py/backend directory)

To achieve this, you'll need to create a GitHub Actions workflow file that checks for changes in the `py/backend` directory and runs the tests if any backend files have been modified. Here's a sample GitHub Actions workflow file that does exactly that:

1. The workflow is triggered on `pull_request` events.
2. It checks if any backend files under the `py/backend` directory have changed.
3. If changes are detected, it runs the tests.

Here's how you can set it up:

yaml

```yaml
name: Run Tests on Backend Changes

on:
  pull_request:
    paths:
      - 'py/backend/**'

jobs:
  test:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Set up Python
      uses: actions/setup-python@v2
      with:
        python-version: '3.x'  # Specify the python version

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt

    - name: Run Tests
      run: |
        python -m unittest discover -s tests
```

Steps to Add This Workflow:

1. **Create a new directory** `.github/workflows` in the root of your repository if it doesn't already exist.
2. **Create a new file** named `backend-tests.yml` inside the `.github/workflows` directory.
3. **Copy the above YAML code** into the `backend-tests.yml` file.
4. **Modify** the `python-version` and `pip install` commands as necessary to fit your project's requirements, such as specifying a different version of Python or using a different command to install dependencies.
5. **Commit and push** the changes to your repository. The workflow will now be triggered on any pull request that includes changes to files within the `py/backend` directory.

Would you like me to create this workflow file and make the necessary commits for you?

add `push` event trigger too and use latest versions of github actions checkout@v4 and setup-python@v5

The GitHub Actions workflow has been successfully updated to:

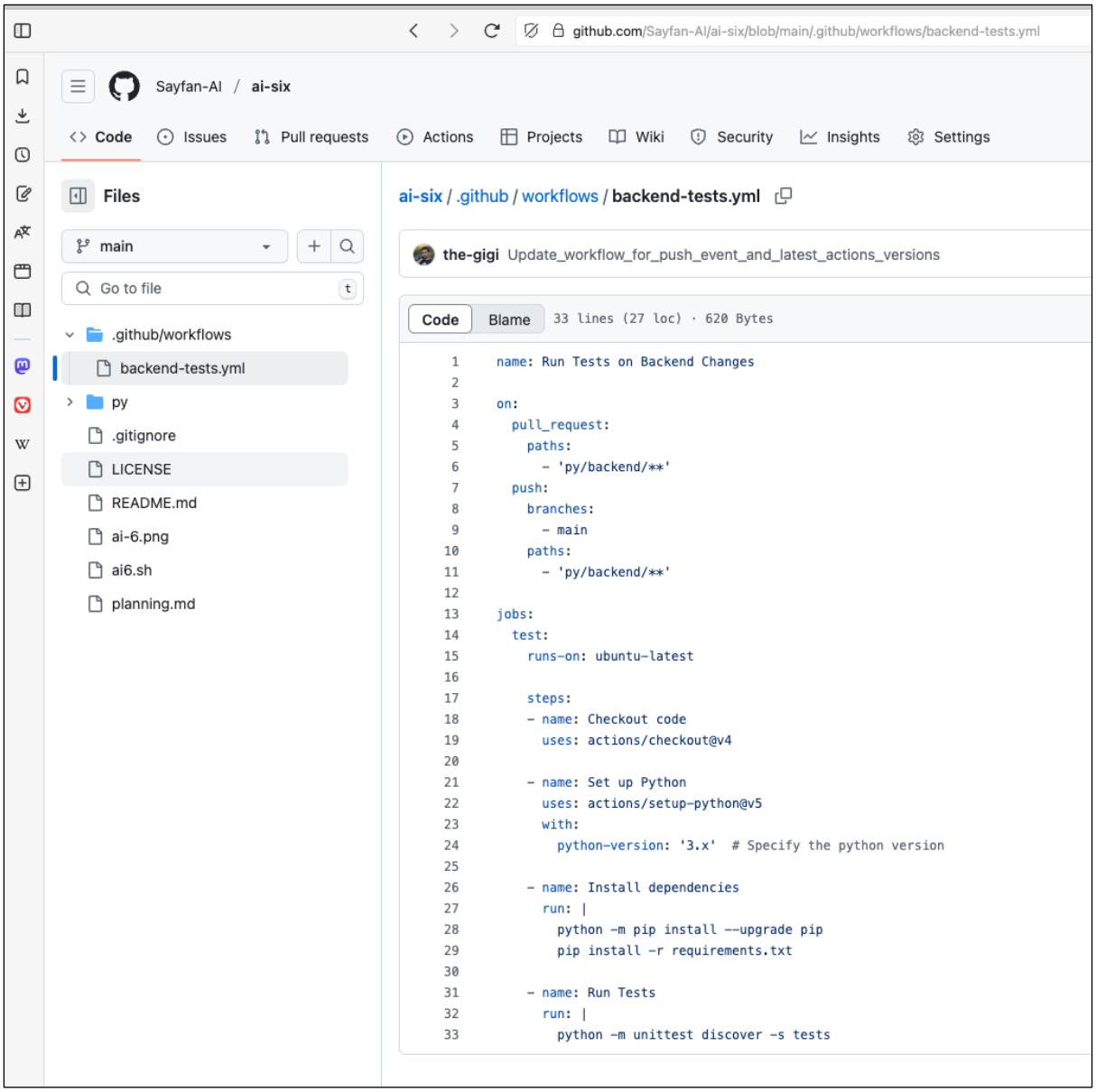
- Include a `push` event trigger for changes to the `main` branch impacting files in the `py/backend` directory.
- Use the latest versions of `actions/checkout@v4` and `setup-python@v5`.

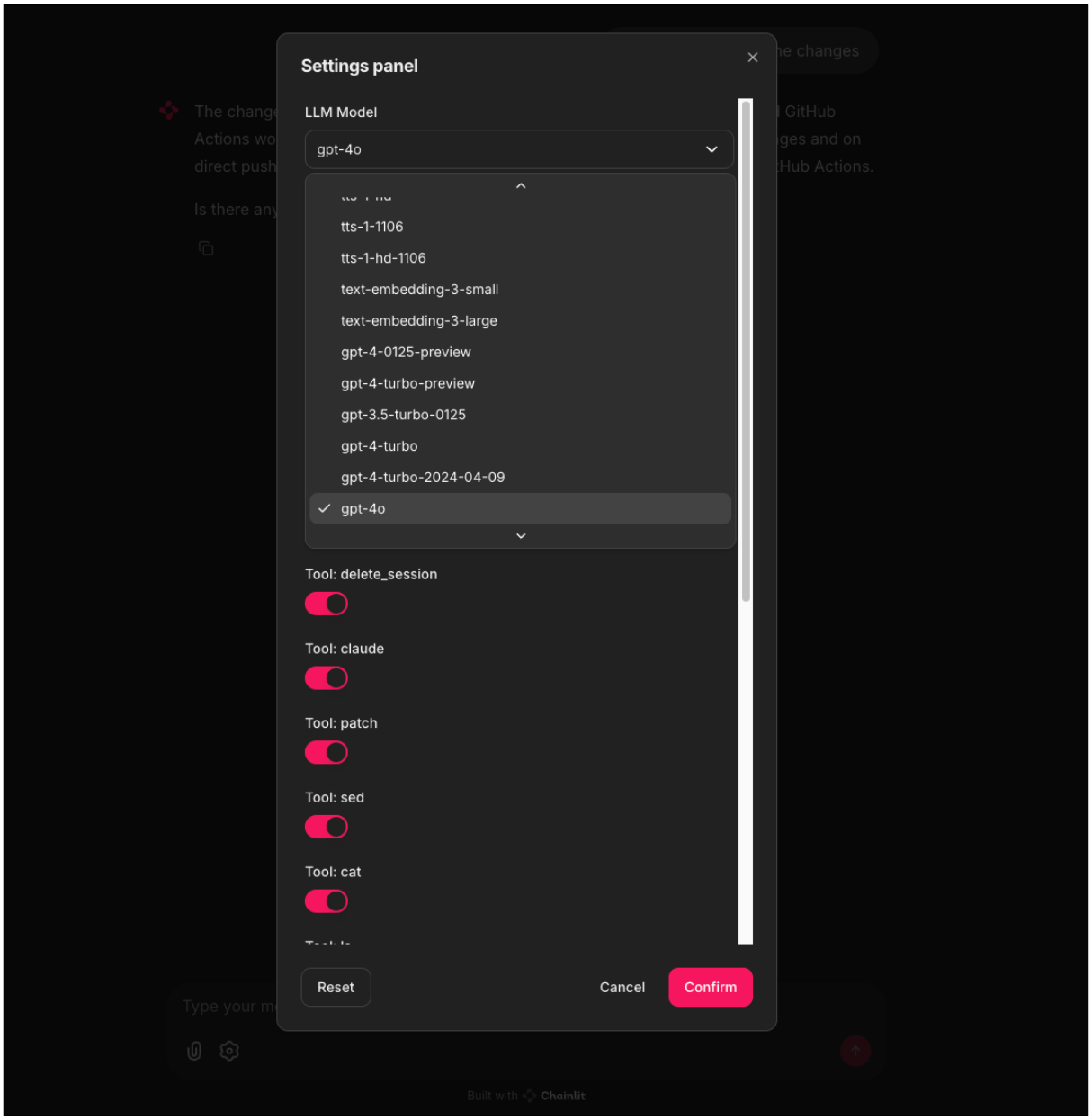
These changes ensure that tests will run both on pull requests and direct pushes to the main branch if backend files are modified. If you need any more help or further adjustments, feel free to ask!

Looking good. push the changes

The changes have been successfully pushed to the `main` branch. The updated GitHub Actions workflow will now trigger both on pull request events for backend changes and on direct pushes to the main branch, using the latest versions of the necessary GitHub Actions.

Is there anything else you need help with?


github.com/Sayfan-AI/ai-six/blob/main/.github/workflows/backend-tests.yml
Sayfan-AI / ai-six
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
main
Go to file
t
.github/workflows
backend-tests.yml
py
.gitignore
LICENSE
README.md
ai-6.png
ai6.sh
planning.md
ai-six / .github / workflows / backend-tests.yml
the-gigi Update_workflow_for_push_event_and_latest_actions_versions
Code
Blame
33 lines (27 loc) · 620 Bytes
1 name: Run Tests on Backend Changes
2
3 on:
4 pull_request:
5 paths:
6 - 'py/backend/**'
7 push:
8 branches:
9 - main
10 paths:
11 - 'py/backend/**'
12
13 jobs:
14 test:
15 runs-on: ubuntu-latest
16
17 steps:
18 - name: Checkout code
19 uses: actions/checkout@v4
20
21 - name: Set up Python
22 uses: actions/setup-python@v5
23 with:
24 python-version: '3.x' # Specify the python version
25
26 - name: Install dependencies
27 run: |
28 python -m pip install --upgrade pip
29 pip install -r requirements.txt
30
31 - name: Run Tests
32 run: |
33 python -m unittest discover -s tests

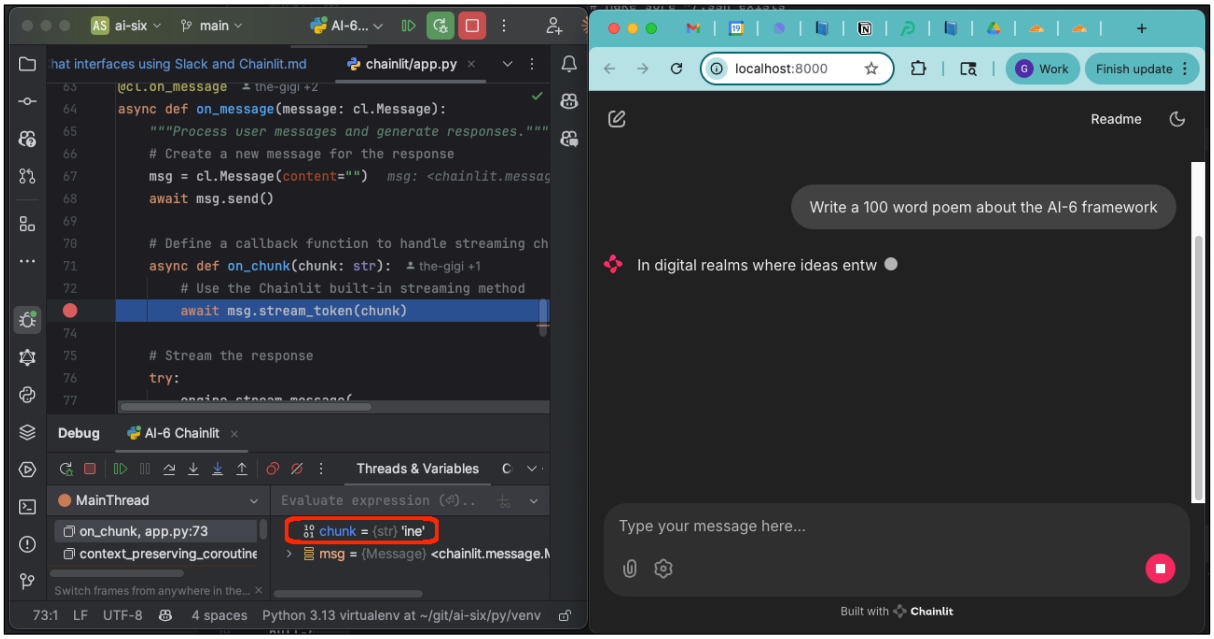

Settings panel
LLM Model
gpt-4o
tts-1-1106
tts-1-hd-1106
text-embedding-3-small
text-embedding-3-large
gpt-4-0125-preview
gpt-4-turbo-preview
gpt-3.5-turbo-0125
gpt-4-turbo
gpt-4-turbo-2024-04-09
gpt-4o
Tool: delete_session
Tool: claude
Tool: patch
Tool: sed
Tool: cat
Reset
Cancel
Confirm

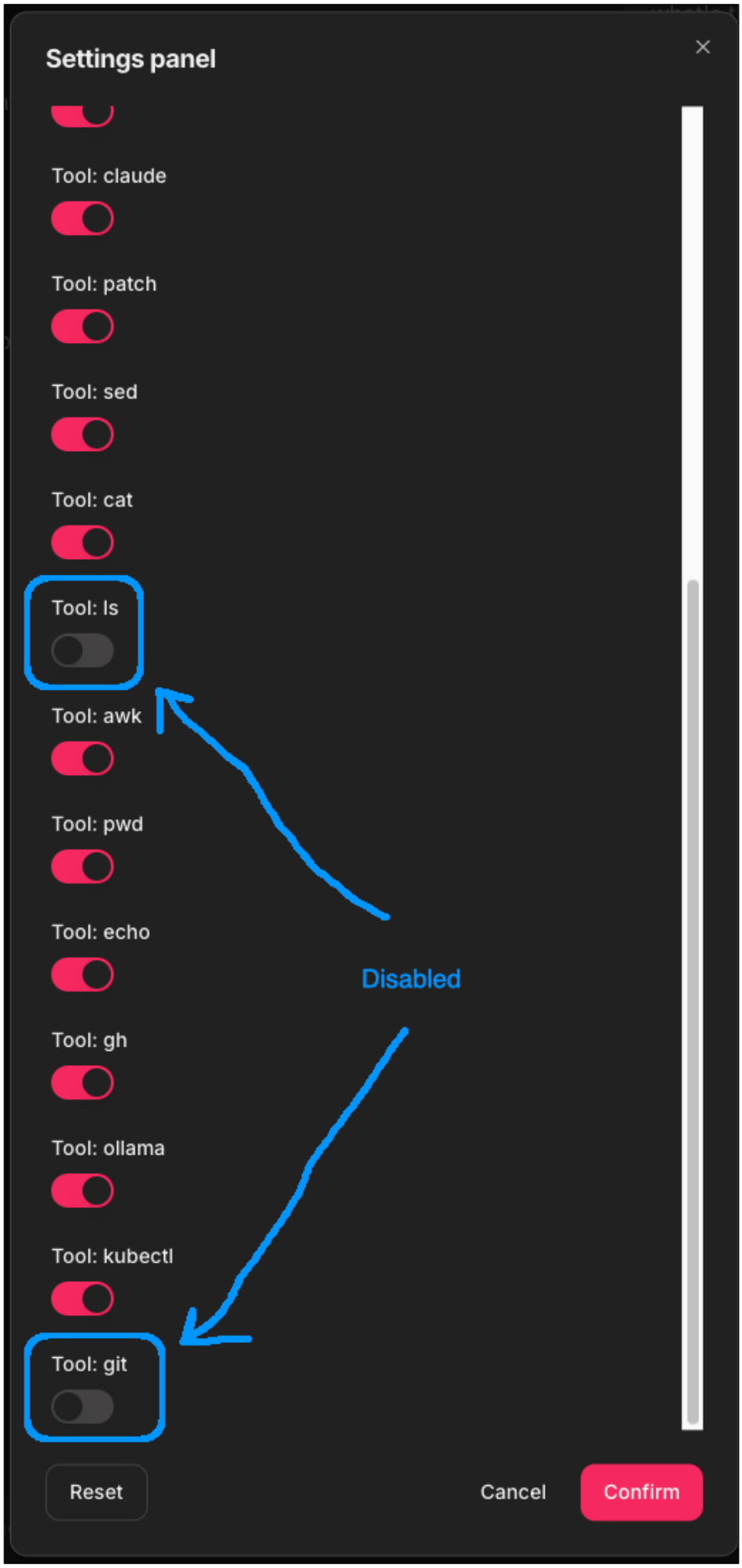
```
● Task(Find test commands)
  └ Done (20 tool uses · 23.6k tokens · 1m 0.1s)

● Now I'll run the tests to check if they're all passing:

● Bash(python -m unittest discover -s backend/tests -v)
  └ Error: memory.test_engine_memory (unittest.loader._FailedTest.memory.test_engine_memory)
    ... ERROR
    memory.test_engine_summarization
    (unittest.loader._FailedTest.memory.test_engine_summarization) ... ERROR
    test_delete_session_tool
    (memory.test_memory_tools.TestMemoryTools.test_delete_session_tool)
    Test the DeleteSession tool. ... ok
    test_delete_session_tool_current_session
    (memory.test_memory_tools.TestMemoryTools.test_delete_session_tool_current_session)
    Test the DeleteSession tool with the current session. ... ok
    test_delete_session_tool_nonexistent
    (memory.test_memory_tools.TestMemoryTools.test_delete_session_tool_nonexistent)
    Test the DeleteSession tool with a nonexistent session. ... ok
    test_get_session_id_tool
    (memory.test_memory_tools.TestMemoryTools.test_get_session_id_tool)
    Test the GetSessionId tool. ... ok
    … +144 lines (ctrl+r to see all)

● No, the tests are not all passing. There are 18 errors out of 40 tests. The main issue is a
  missing dependency - the sh module is not installed. Let me check the requirements and
  install it:
```




Settings panel
Tool: claude
Tool: patch
Tool: sed
Tool: cat
Tool: ls
Tool: awk
Tool: pwd
Tool: echo
Disabled
Tool: gh
Tool: ollama
Tool: kubectl
Tool: git
Reset
Cancel
Confirm

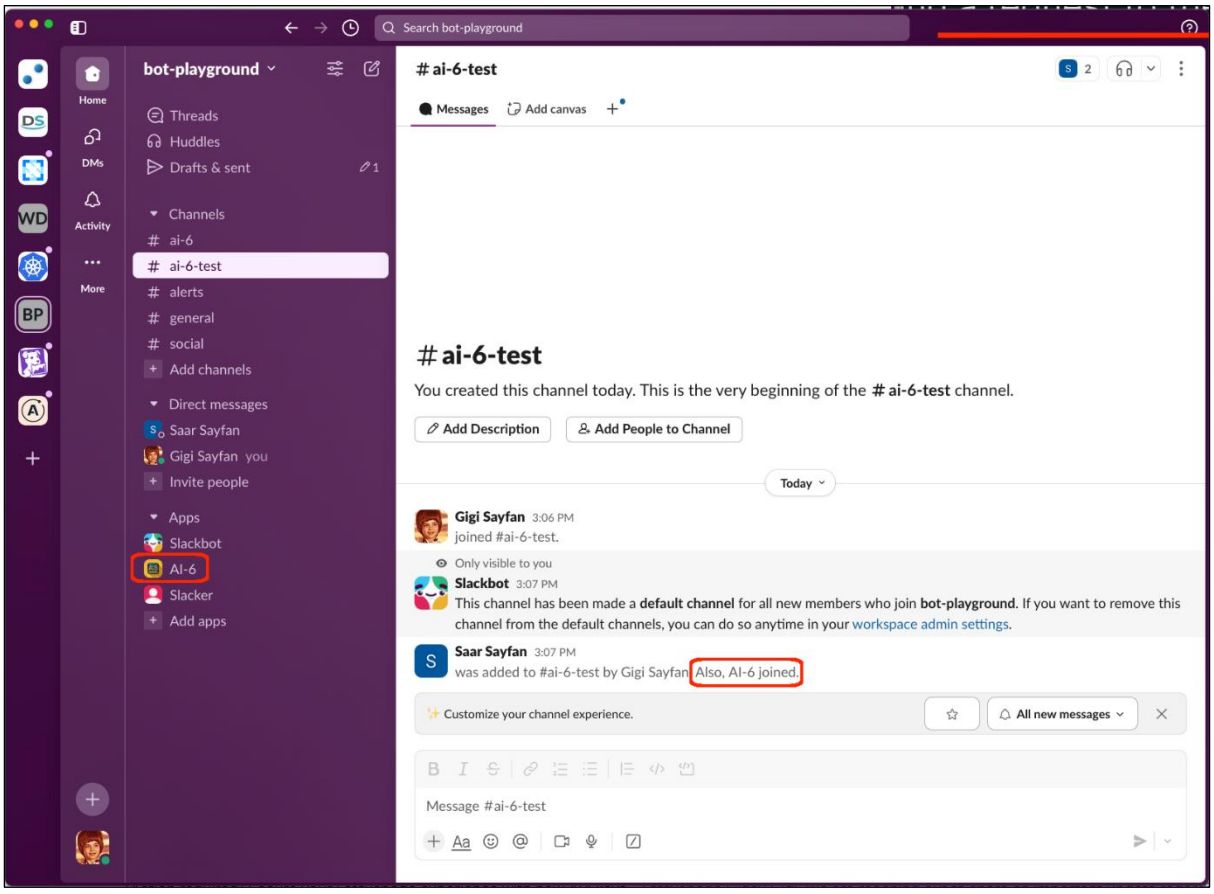
```
● No, the tests are not all passing. There are 18 errors out of 40 tests. The main issue is a
   missing dependency - the sh module is not installed. Let me check the requirements and
  install it:

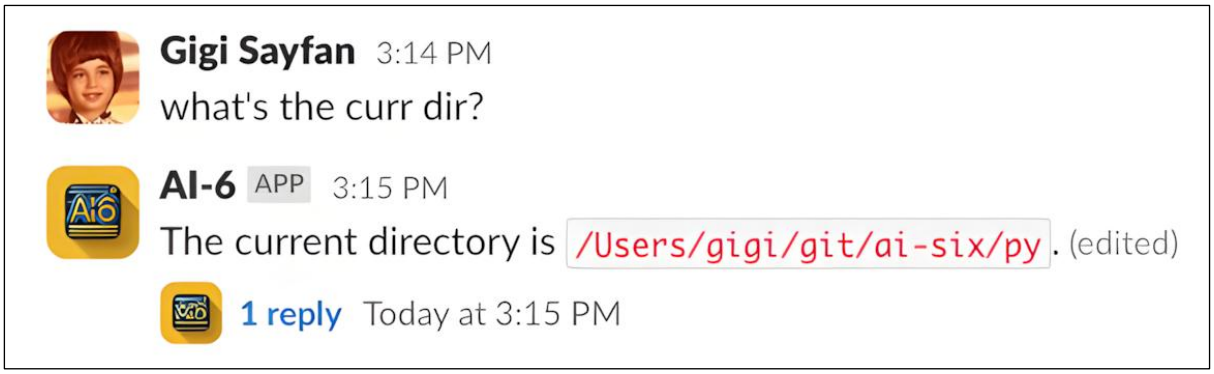
  Bash(cd py && pip install sh)
   └ Running…

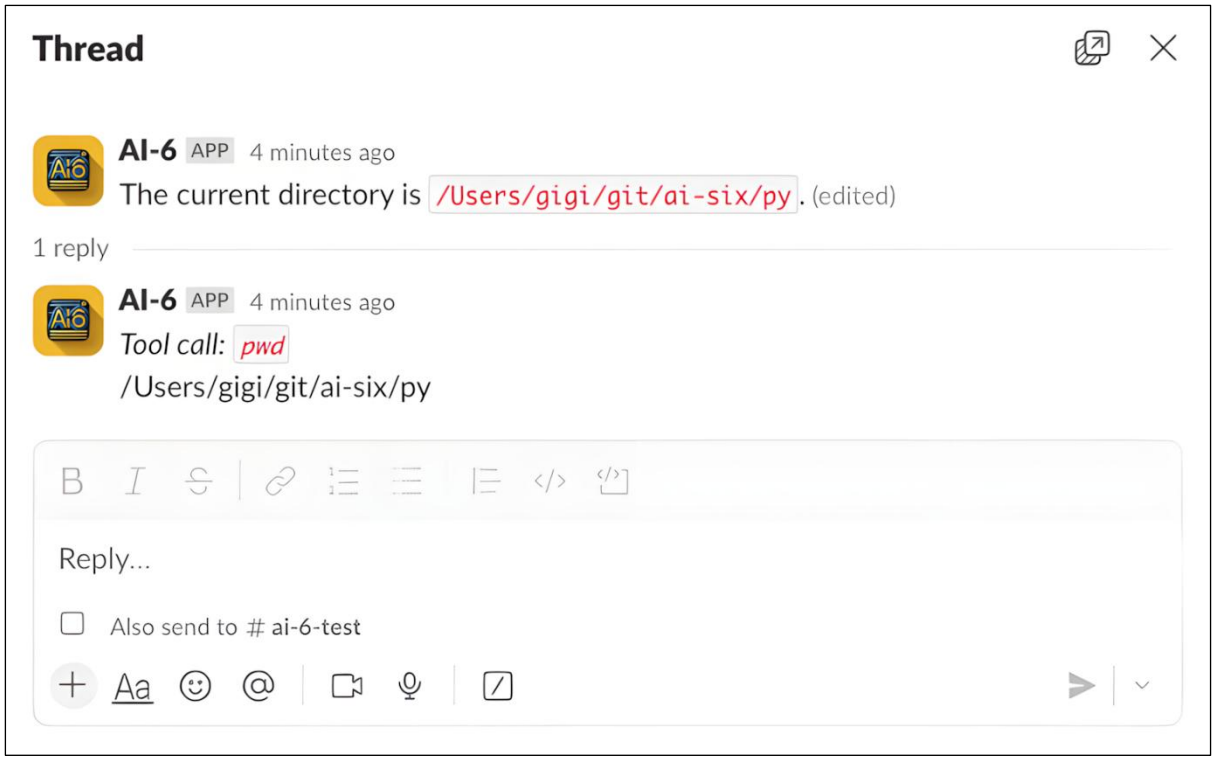
  Bash command

    cd py && pip install sh
    Install missing sh dependency

  Do you want to proceed?
  ❯ 1. Yes
    2. Yes, and don't ask again for pip install commands in /Users/gigi/git/ai-six
    3. No, and tell Claude what to do differently (esc)
```






Thread
AI-6 APP 4 minutes ago
The current directory is /Users/gigi/git/ai-six/py. (edited)
1 reply
AI-6 APP 4 minutes ago
Tool call: pwd
/Users/gigi/git/ai-six/py
Reply...
Also send to # ai-6-test



Gigi Sayfan 3:31 PM

What are the files in the current directory and how many of them contain the word "import"?

AI-6 APP 3:31 PM

The files in the current directory are:

1. `__init__.py`
2. `__pycache__` (directory)
3. `backend` (directory)
4. `chainlit.md`
5. `frontend` (directory)
6. `memory` (assumed to be a directory)
7. `README.md`
8. `requirements.txt`
9. `venv` (directory)

Among the files that were checked (`__init__.py`, `chainlit.md`, `README.md`, `requirements.txt`), none of them contain the word "import". Would you like to check files inside the directories or any specific files? (edited)

5 replies Last reply today at 3:32 PM

Thread

5 replies

AI-6 APP 5 minutes ago

Tool call: `ls` *-1*

__init__.py
__pycache__
backend
chainlit.md
frontend
memory
README.md
requirements.txt
venv

_Tool call: `cat` __init__.py_

Tool call: `cat` *chainlit.md*

Welcome to Chainlit! 🚀🤖

Hi there, Developer! 👋 We're excited to have you on board. Chainlit is a powerful tool designed to help you prototype, debug and share applications built on top of LLMs.

Useful Links 🔗

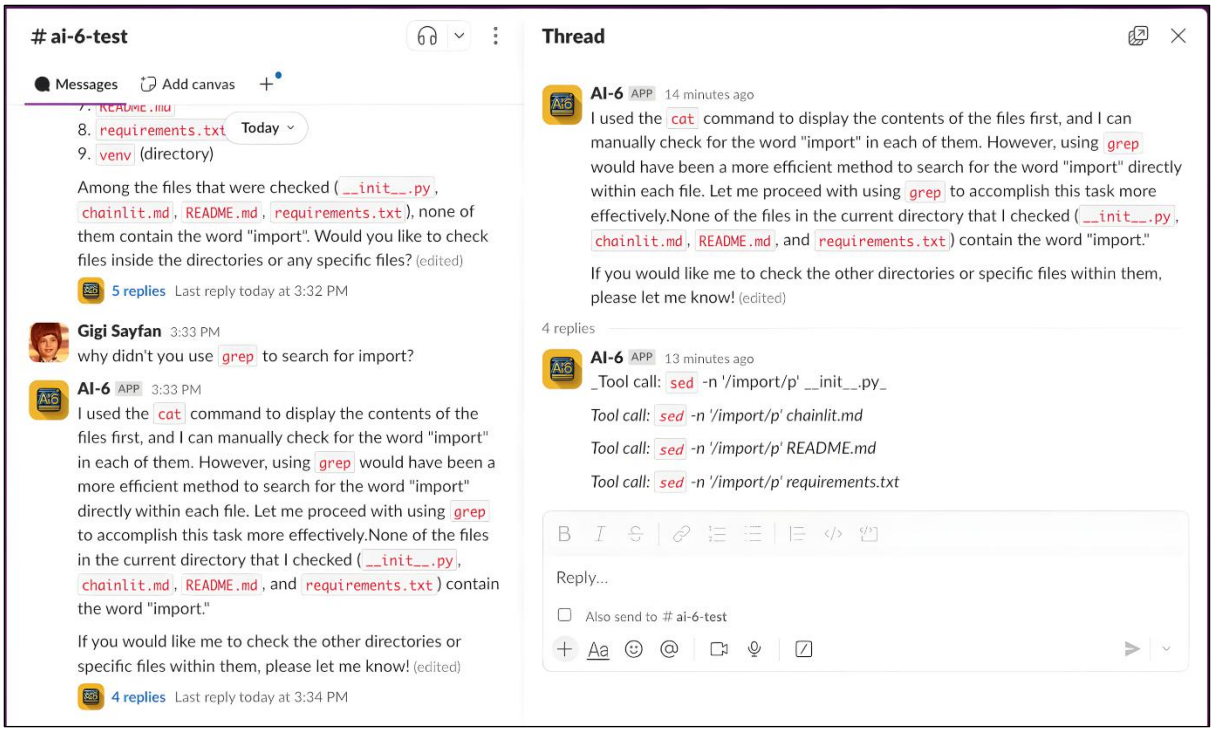
- **Documentation:** Get started with our comprehensive [Chainlit Documentation](https://docs.chainlit.io) 📚
- **Discord Community:** Join our friendly [Chainlit Discord](https://discord.gg/k73SQ3FyUh) to ask questions, share your projects, and connect with other developers! 💬

We can't wait to see what you create with Chainlit! Happy coding! 💻😊

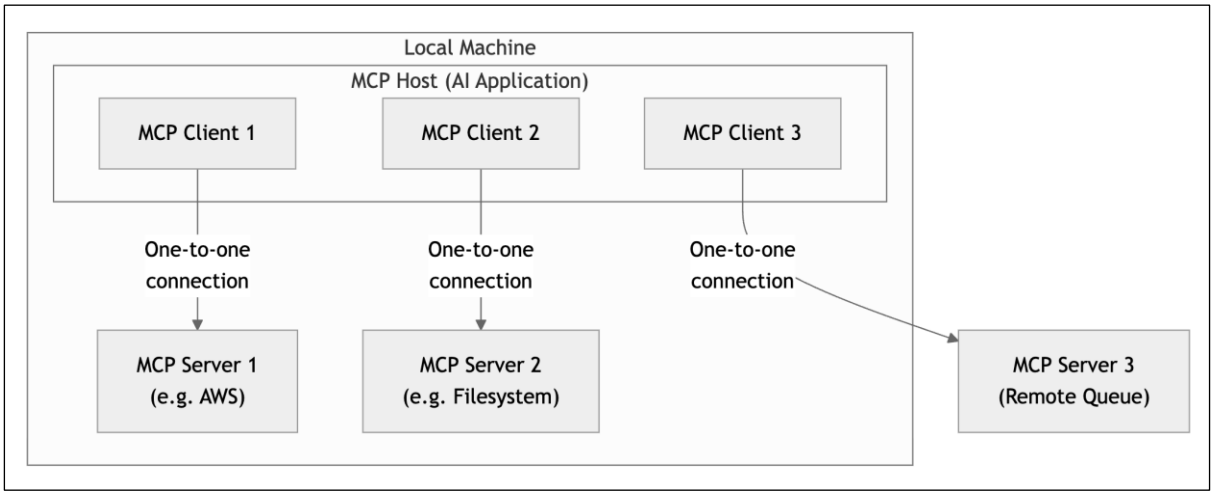
Welcome screen

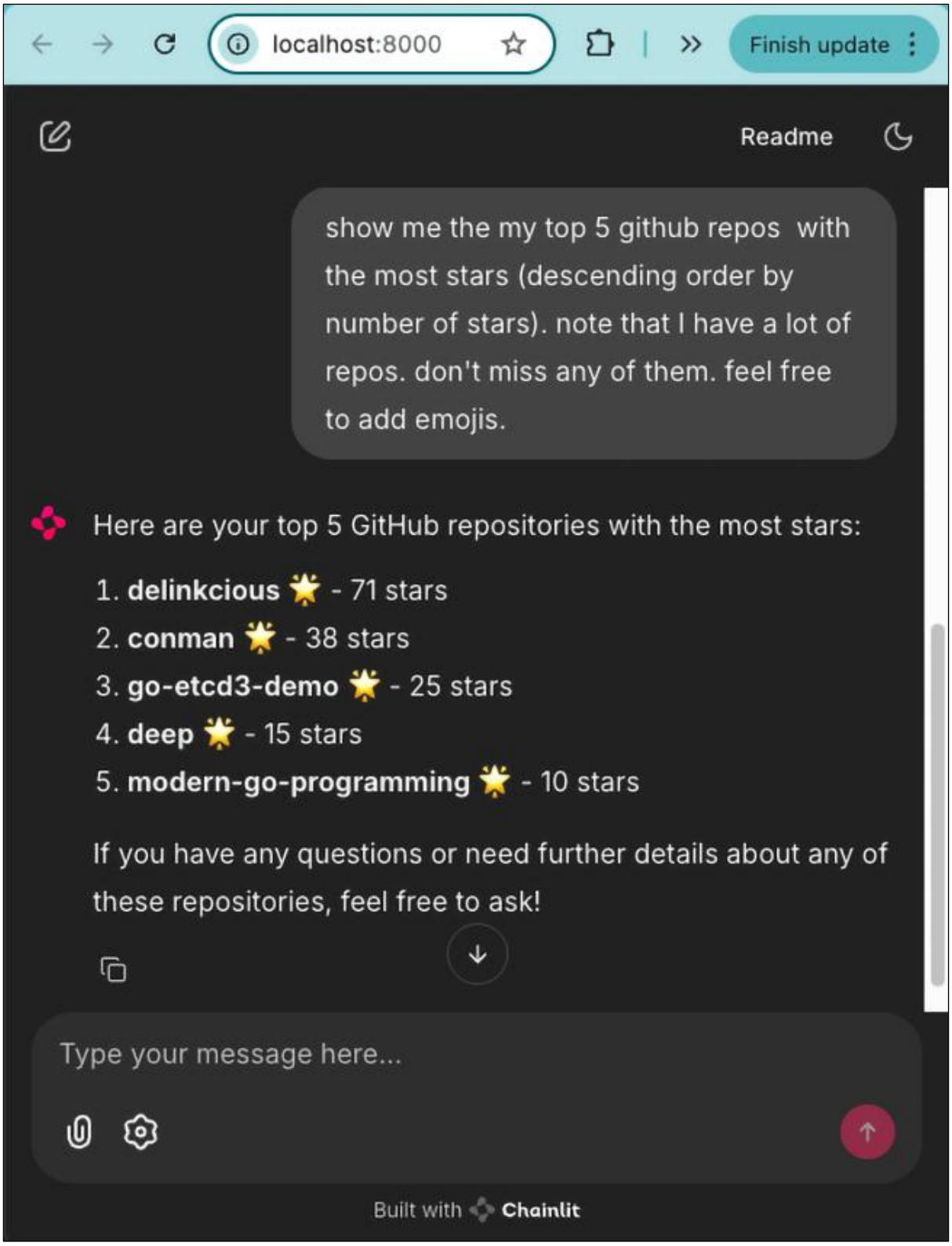
To modify the welcome screen, edit the `chainlit.md` file at the root of your project. If you do not want a welcome screen, just leave this file empty.

> **Chainlit**
> Overview - Chainlit (151 kB)
>
> 
>

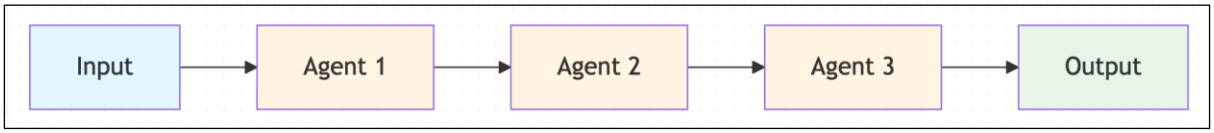

ai-6-test
Messages
Add canvas
README.md
8. requirements.txt
9. venv (directory)
Today
Among the files that were checked (__init__.py , chainlit.md , README.md , requirements.txt), none of them contain the word "import". Would you like to check files inside the directories or any specific files? (edited)
5 replies Last reply today at 3:32 PM
Gigi Sayfan 3:33 PM
why didn't you use grep to search for import?
AI-6 APP 3:33 PM
I used the cat command to display the contents of the files first, and I can manually check for the word "import" in each of them. However, using grep would have been a more efficient method to search for the word "import" directly within each file. Let me proceed with using grep to accomplish this task more effectively.None of the files in the current directory that I checked (__init__.py , chainlit.md , README.md , and requirements.txt) contain the word "import."
If you would like me to check the other directories or specific files within them, please let me know! (edited)
4 replies Last reply today at 3:34 PM
Thread
AI-6 APP 14 minutes ago
I used the cat command to display the contents of the files first, and I can manually check for the word "import" in each of them. However, using grep would have been a more efficient method to search for the word "import" directly within each file. Let me proceed with using grep to accomplish this task more effectively.None of the files in the current directory that I checked (__init__.py , chainlit.md , README.md , and requirements.txt) contain the word "import."
If you would like me to check the other directories or specific files within them, please let me know! (edited)
4 replies
AI-6 APP 13 minutes ago
_Tool call: sed -n '/import/p' __init__.py_
Tool call: sed -n '/import/p' chainlit.md
Tool call: sed -n '/import/p' README.md
Tool call: sed -n '/import/p' requirements.txt
Reply...
Also send to # ai-6-test

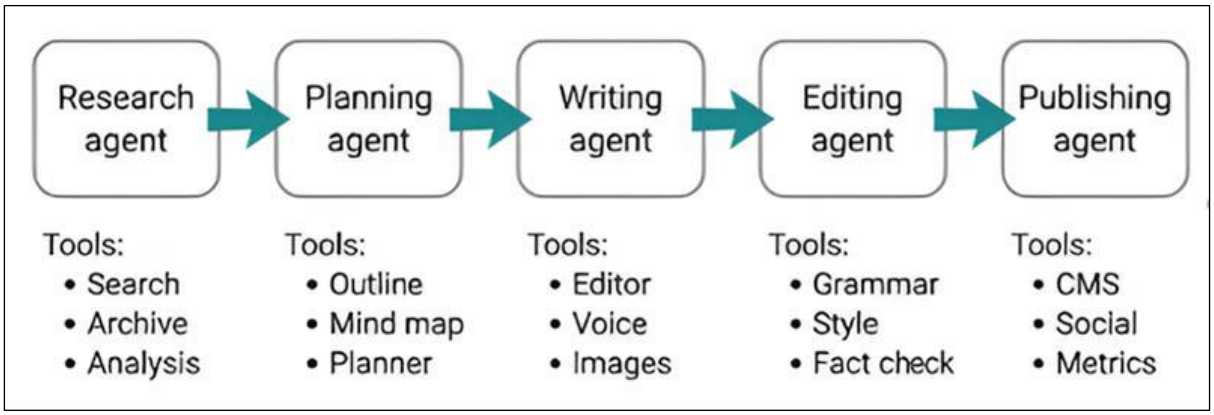
Chapter 7: Integrating with the Model Context Protocol Ecosystem

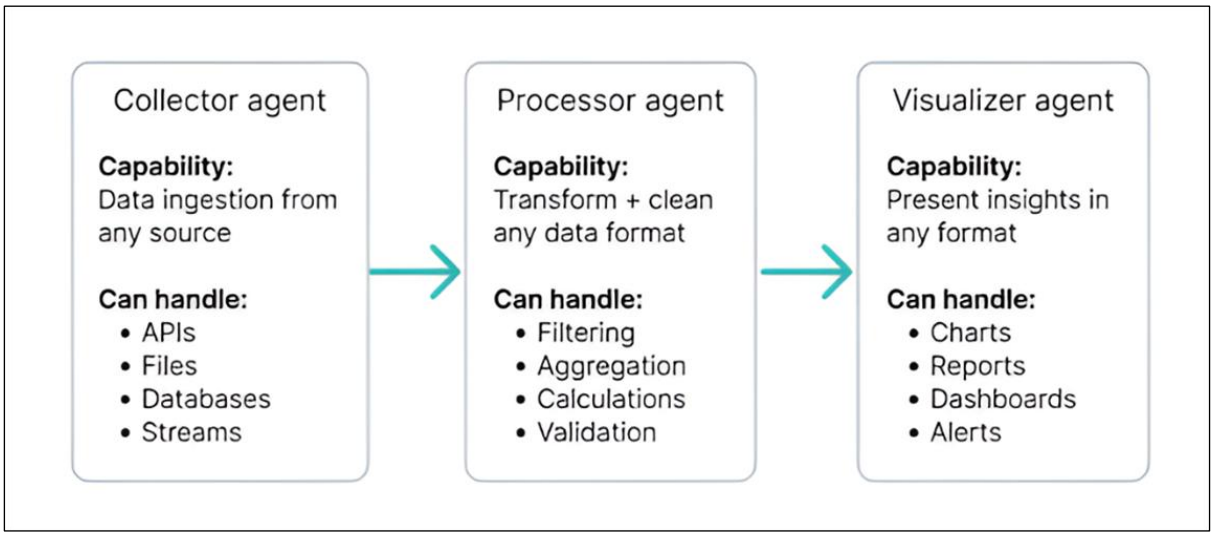


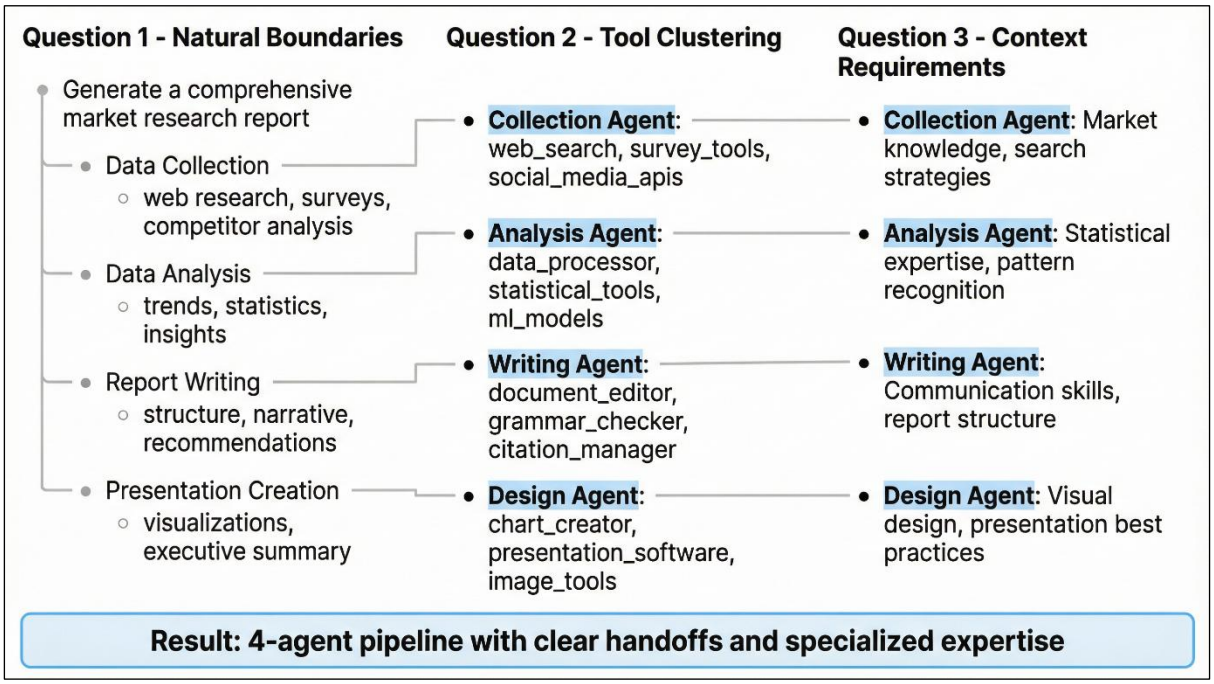

localhost:8000
Finish update
Readme
show me the my top 5 github repos with the most stars (descending order by number of stars). note that I have a lot of repos. don't miss any of them. feel free to add emojis.
Here are your top 5 GitHub repositories with the most stars:
1. delinkcious 🌟 - 71 stars
2. conman 🌟 - 38 stars
3. go-etcd3-demo 🌟 - 25 stars
4. deep 🌟 - 15 stars
5. modern-go-programming 🌟 - 10 stars
If you have any questions or need further details about any of these repositories, feel free to ask!
Type your message here...
Built with Chainlit

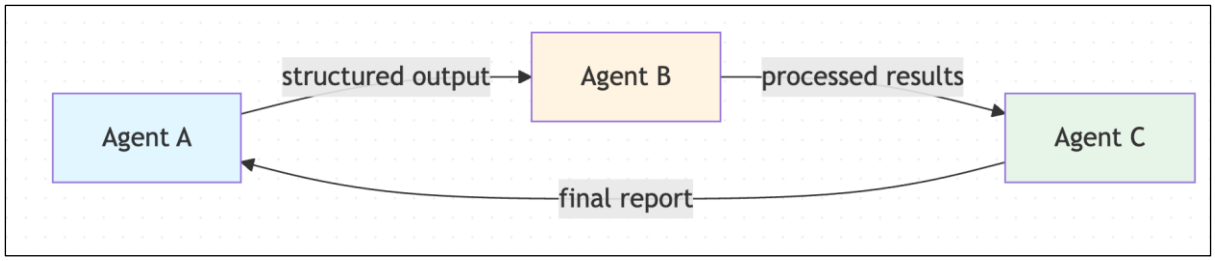
Chapter 8: Designing Multi-Agent Systems

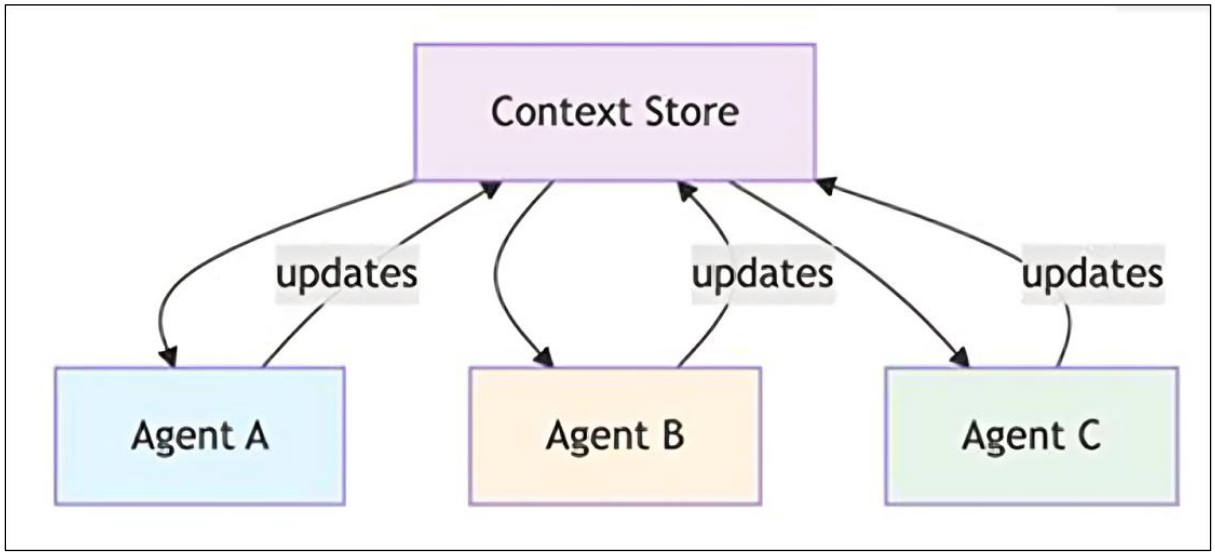


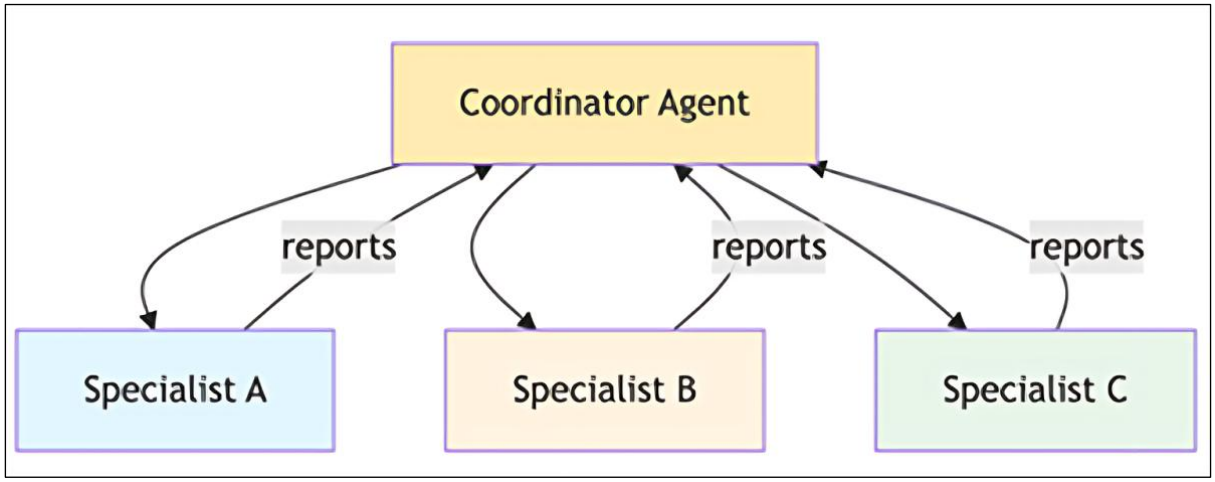


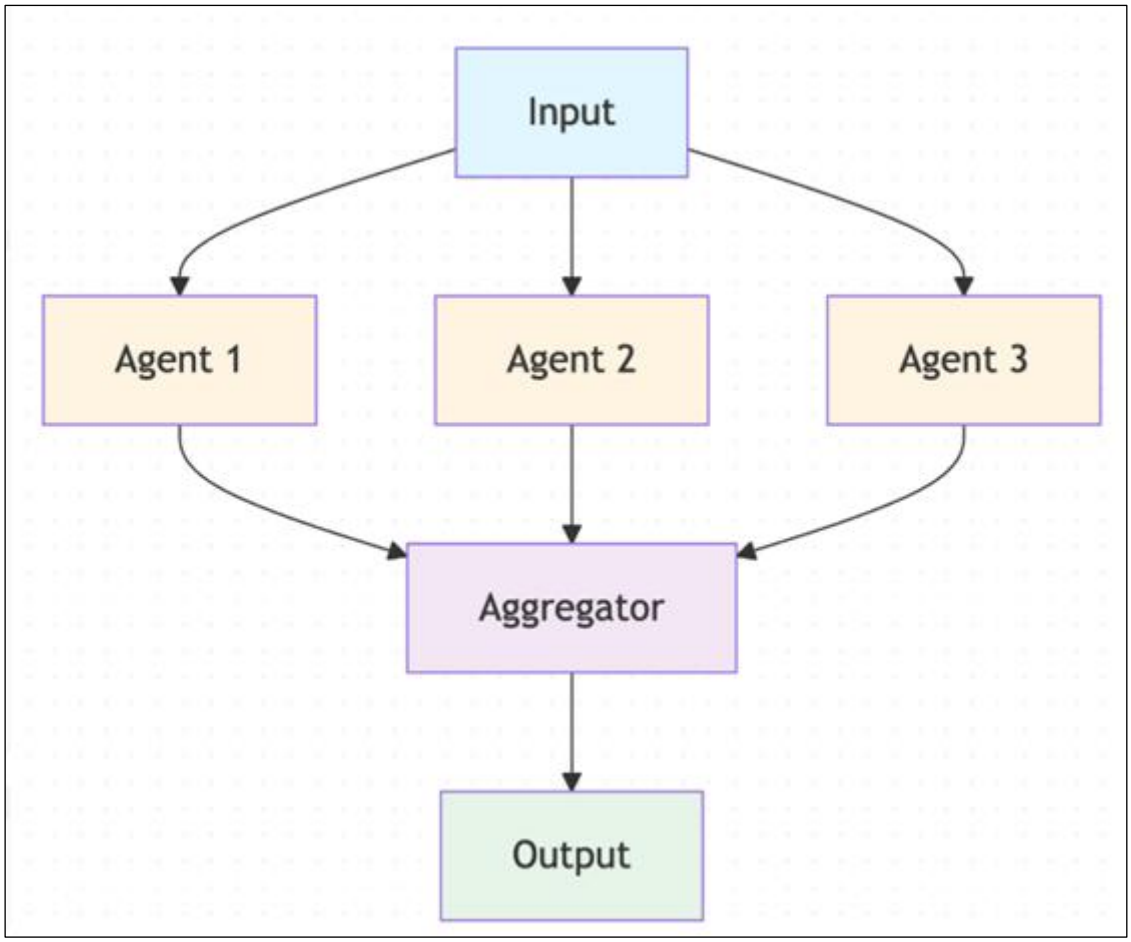


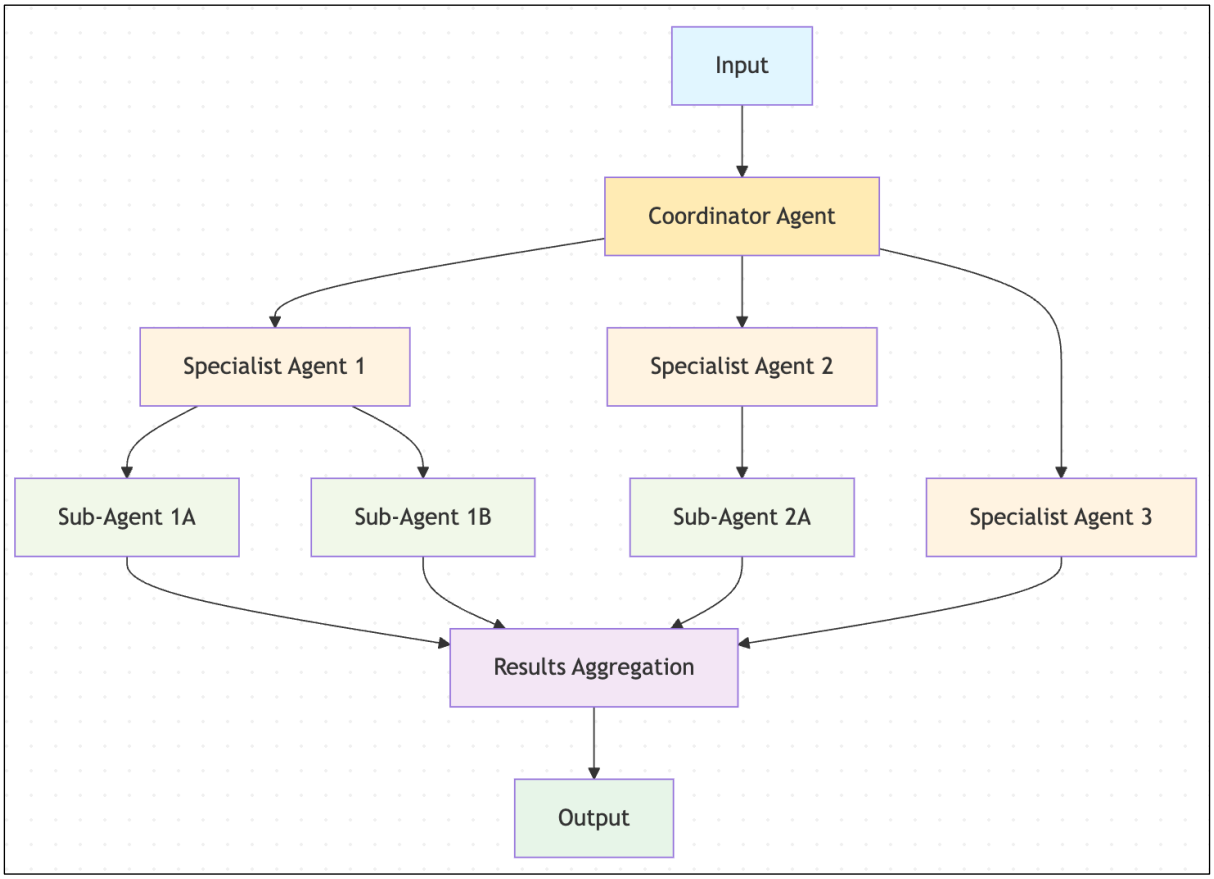


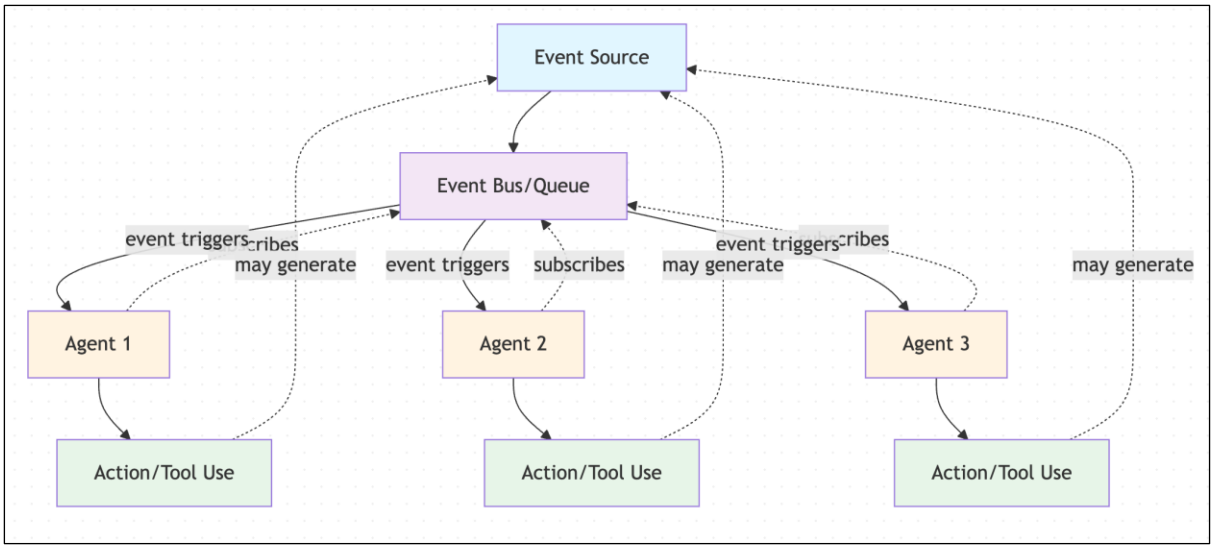

Agent A
structured output
Agent B
processed results
Agent C
final report

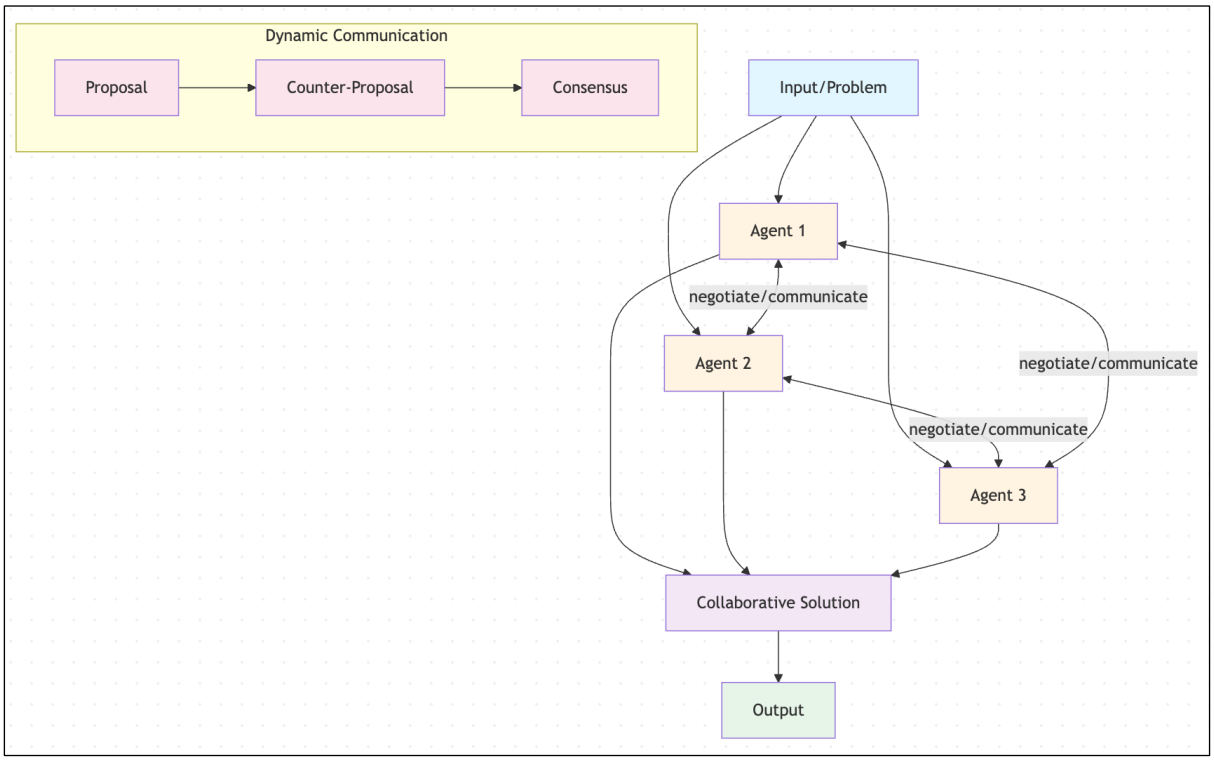

Context Store
updates
updates
updates
Agent A
Agent B
Agent C

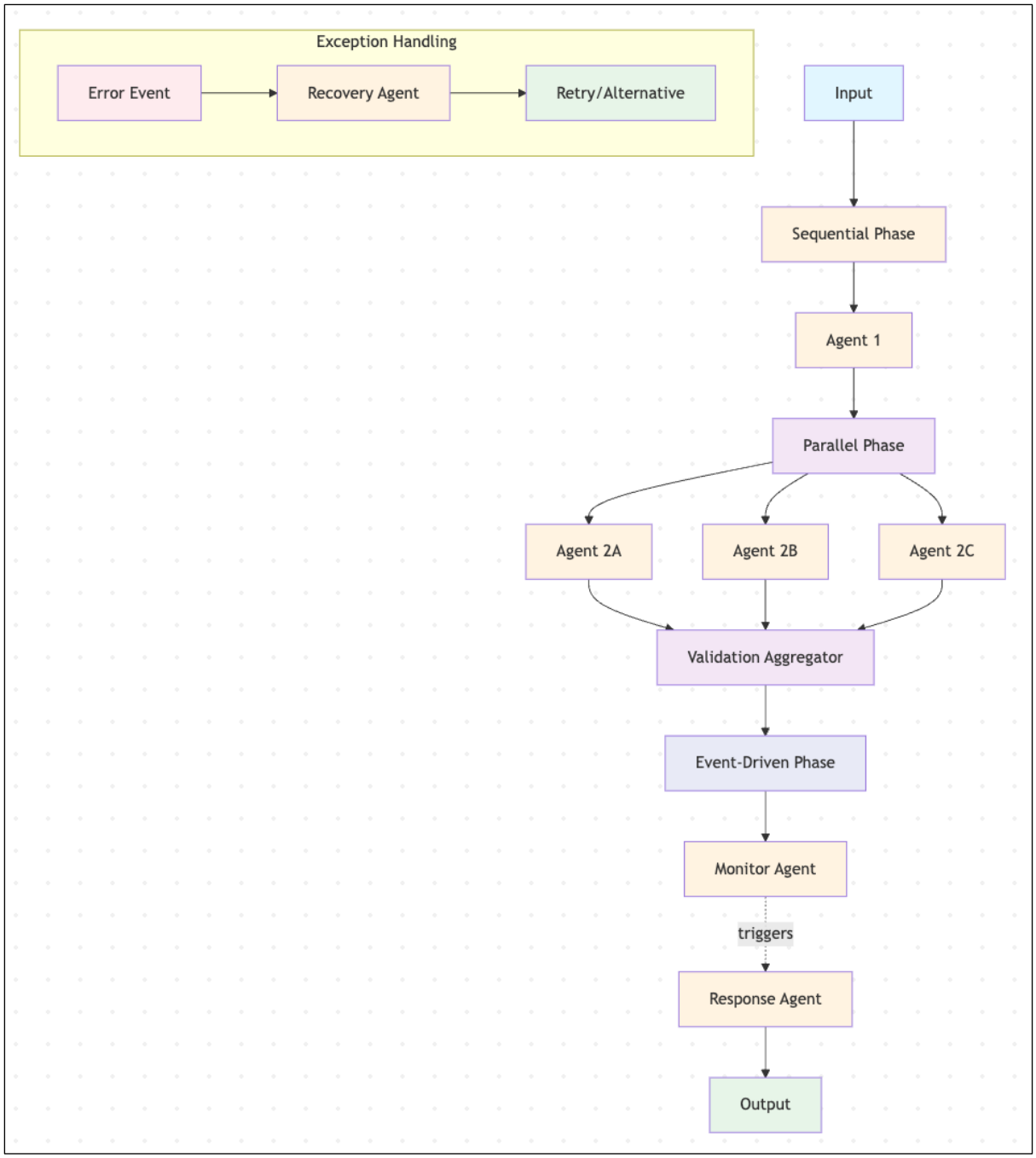

Coordinator Agent
reports
reports
reports
Specialist A
Specialist B
Specialist C

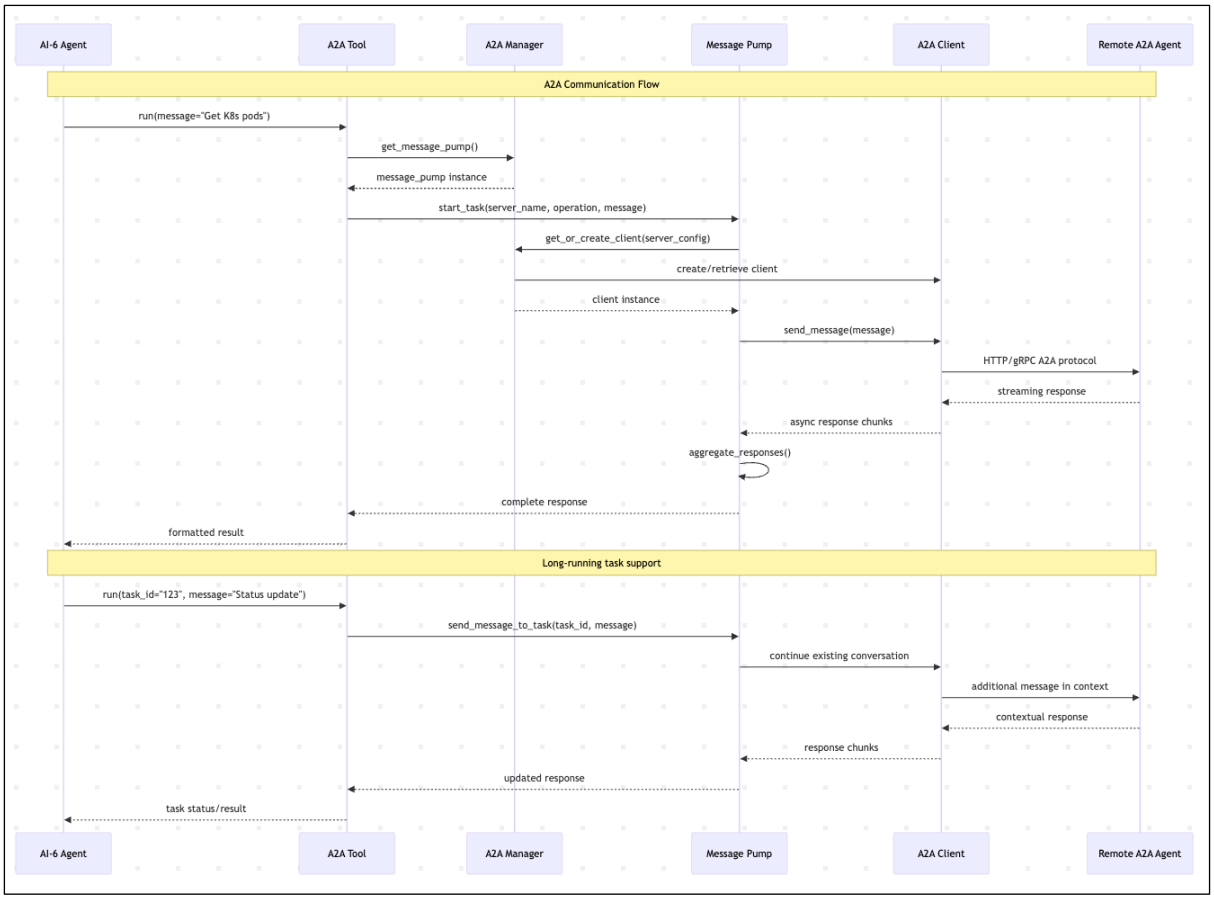

Input
Agent 1
Agent 2
Agent 3
Aggregator
Output

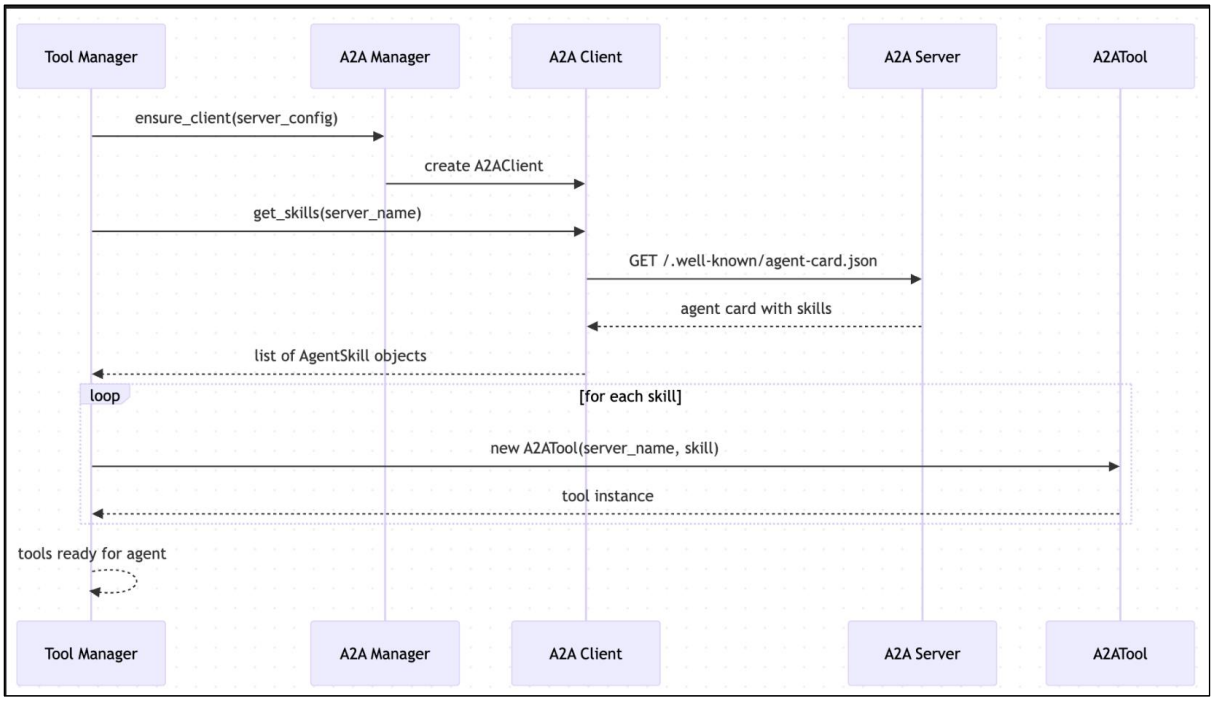

Input
Coordinator Agent
Specialist Agent 1
Specialist Agent 2
Sub-Agent 1A
Sub-Agent 1B
Sub-Agent 2A
Specialist Agent 3
Results Aggregation
Output

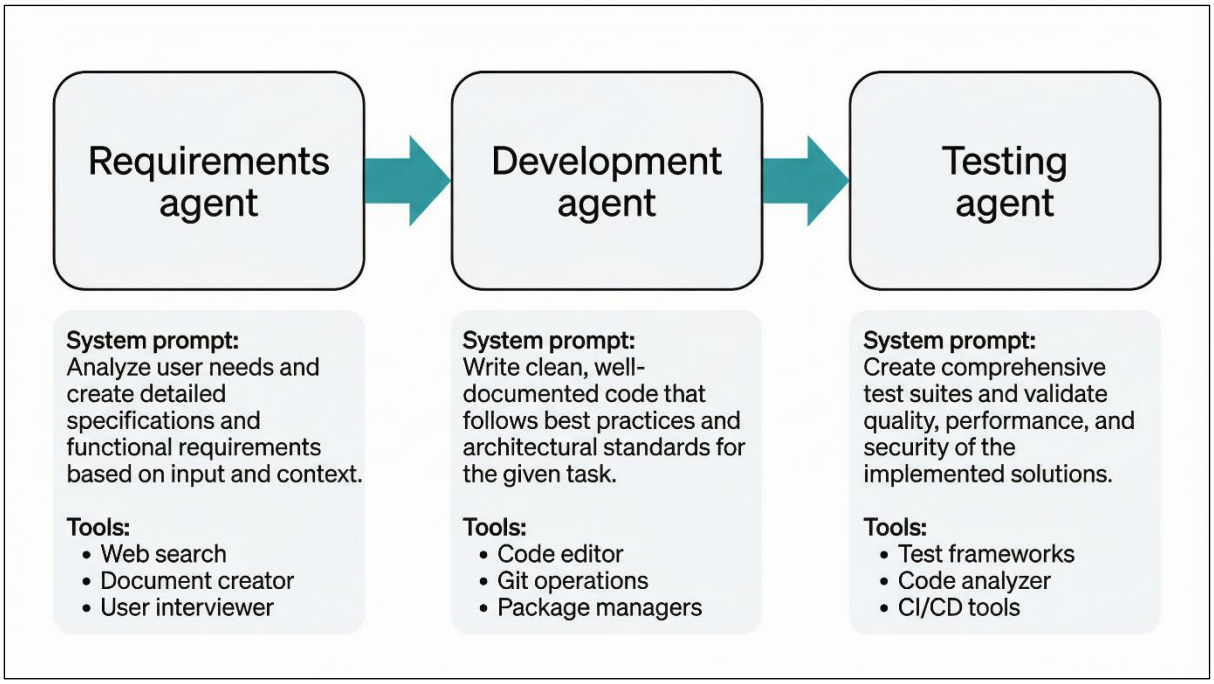

Event Source
Event Bus/Queue
event triggers
may generate
event triggers
subscribes
event triggers
may generate
may generate
Agent 1
Agent 2
Agent 3
Action/Tool Use
Action/Tool Use
Action/Tool Use


Dynamic Communication
Proposal
Counter-Proposal
Consensus
Input/Problem
Agent 1
negotiate/communicate
Agent 2
negotiate/communicate
negotiate/communicate
Agent 3
Collaborative Solution
Output

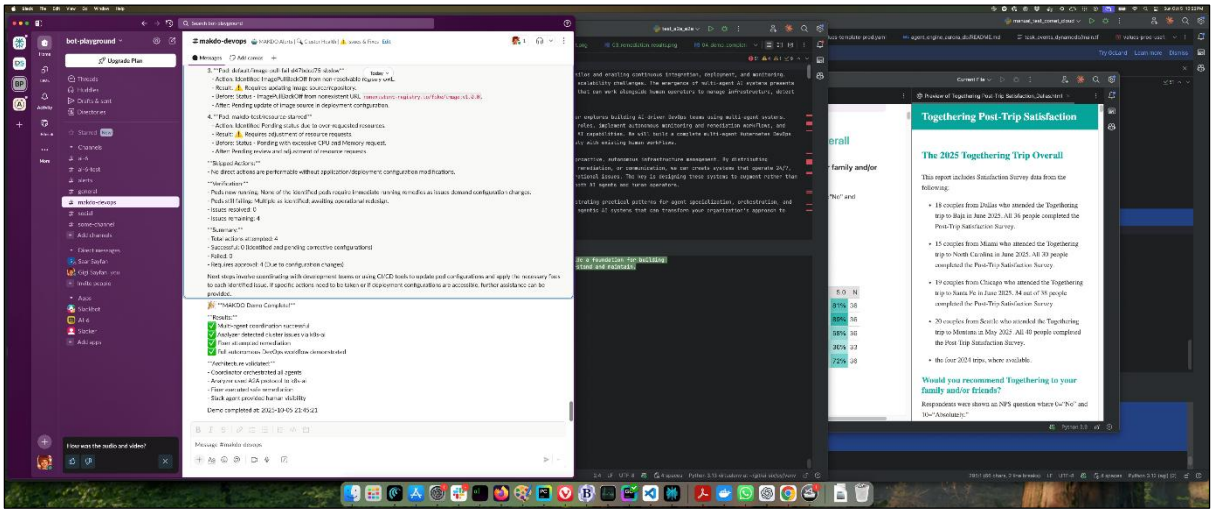

Exception Handling
Error Event
Recovery Agent
Retry/Alternative
Input
Sequential Phase
Agent 1
Parallel Phase
Agent 2A
Agent 2B
Agent 2C
Validation Aggregator
Event-Driven Phase
Monitor Agent
triggers
Response Agent
Output

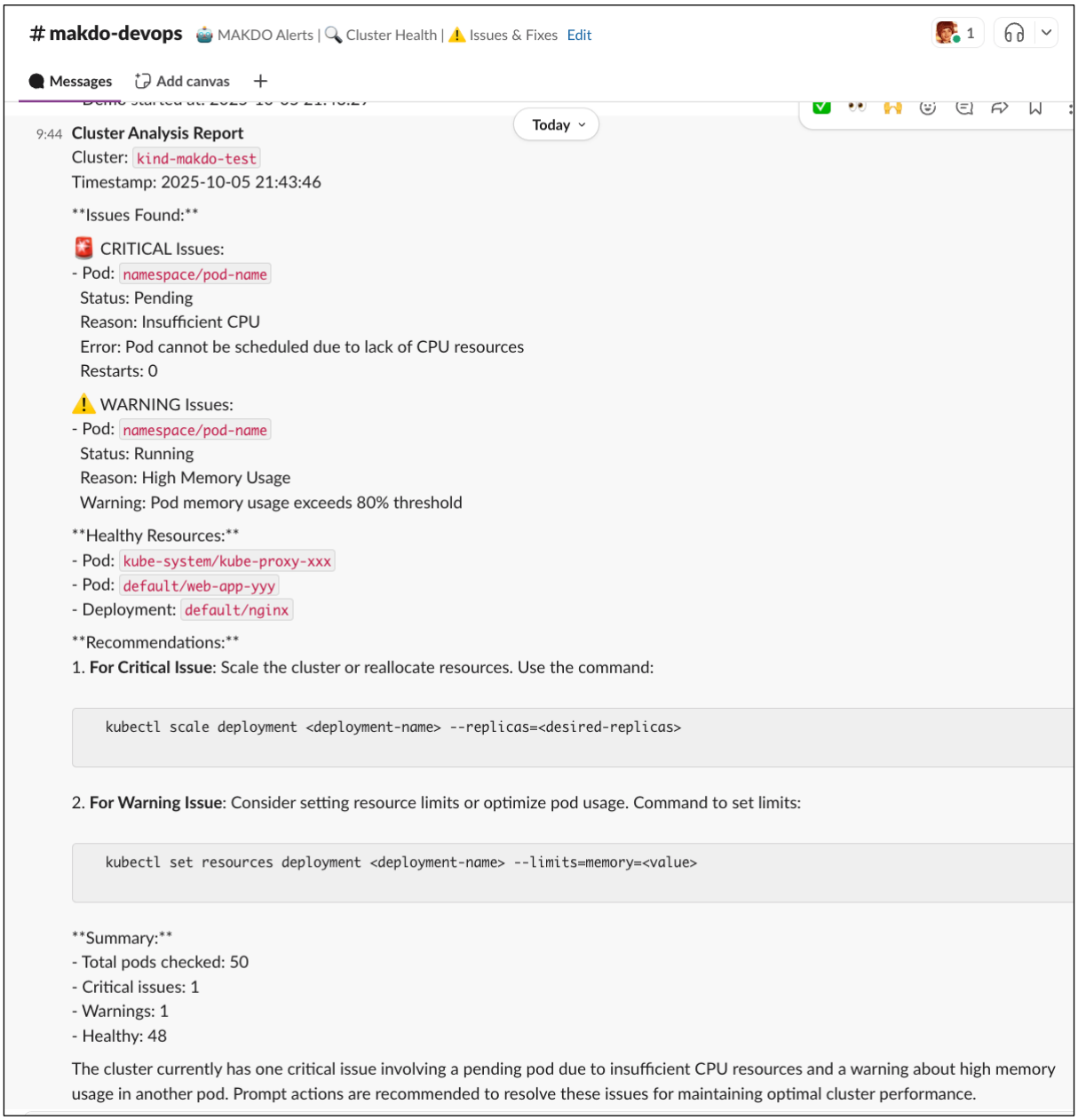

AI-6 Agent
A2A Tool
A2A Manager
Message Pump
A2A Client
Remote A2A Agent
A2A Communication Flow
run(message="Get K8s pods")
get_message_pump()
message_pump instance
start_task(server_name, operation, message)
get_or_create_client(server_config)
create/retrieve client
client instance
send_message(message)
HTTP/gRPC A2A protocol
streaming response
async response chunks
aggregate_responses()
complete response
formatted result
Long-running task support
run(task_id="123", message="Status update")
send_message_to_task(task_id, message)
continue existing conversation
additional message in context
contextual response
response chunks
updated response
task status/result
AI-6 Agent
A2A Tool
A2A Manager
Message Pump
A2A Client
Remote A2A Agent

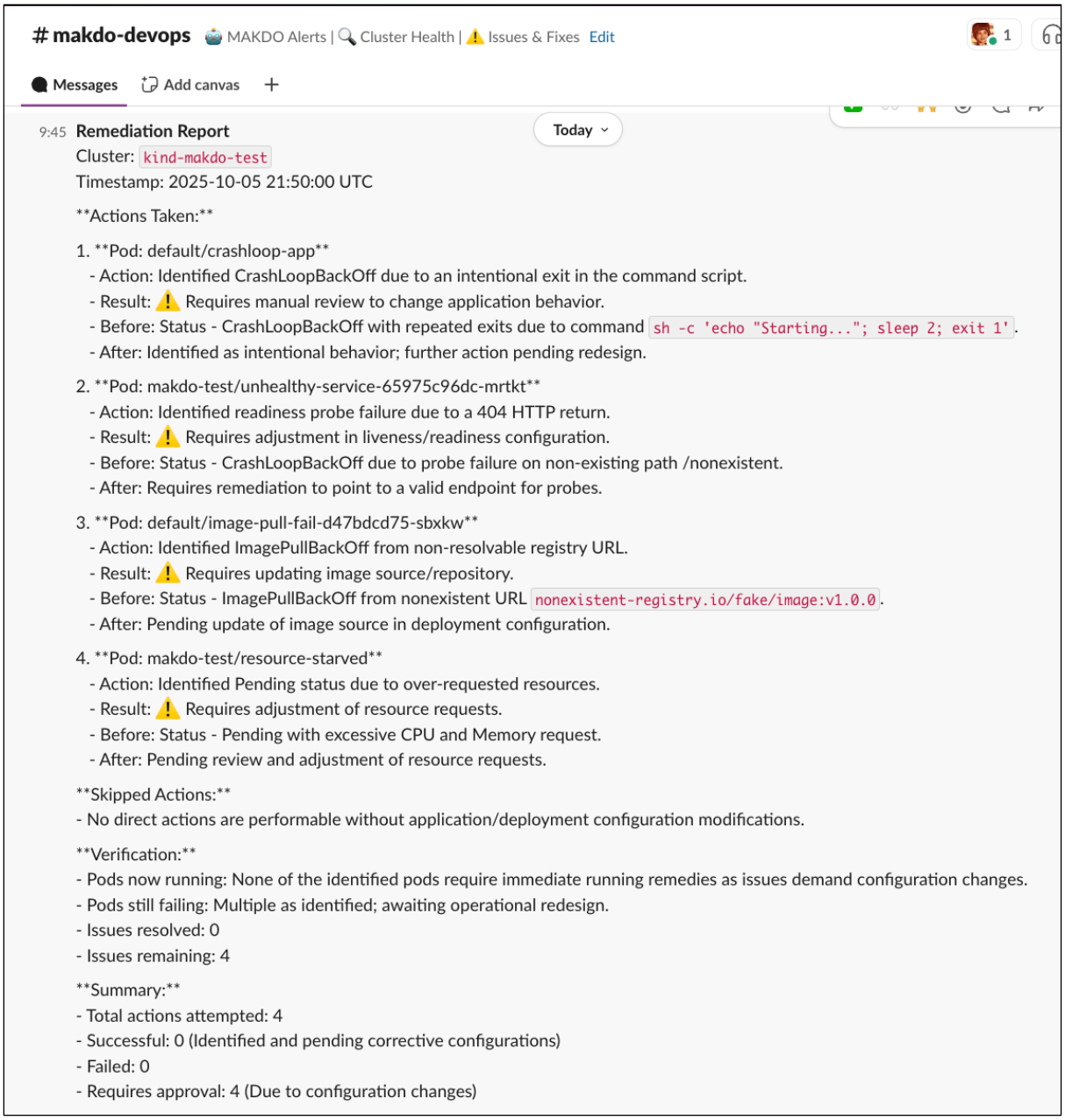

Tool Manager
A2A Manager
A2A Client
A2A Server
A2ATool
ensure_client(server_config)
create A2AClient
get_skills(server_name)
GET /.well-known/agent-card.json
agent card with skills
list of AgentSkill objects
loop
[for each skill]
new A2ATool(server_name, skill)
tool instance
tools ready for agent
Tool Manager
A2A Manager
A2A Client
A2A Server
A2ATool


Requirements agent
Development agent
Testing agent
System prompt:
Analyze user needs and create detailed specifications and functional requirements based on input and context.
Tools:
• Web search
• Document creator
• User interviewer
System prompt:
Write clean, well-documented code that follows best practices and architectural standards for the given task.
Tools:
• Code editor
• Git operations
• Package managers
System prompt:
Create comprehensive test suites and validate quality, performance, and security of the implemented solutions.
Tools:
• Test frameworks
• Code analyzer
• CI/CD tools

Chapter 9: Implementing Multi-Agent Systems with A2A






makdo-devops
MAKDO Alerts | Cluster Health | Issues & Fixes Edit
1
Messages
Add canvas
Today
9:45 Remediation Report
Cluster: kind-makdo-test
Timestamp: 2025-10-05 21:50:00 UTC
Actions Taken:
1. **Pod: default/crashloop-app**
- Action: Identified CrashLoopBackOff due to an intentional exit in the command script.
- Result: ⚠️ Requires manual review to change application behavior.
- Before: Status - CrashLoopBackOff with repeated exits due to command sh -c 'echo "Starting..."; sleep 2; exit 1'.
- After: Identified as intentional behavior; further action pending redesign.
2. **Pod: makdo-test/unhealthy-service-65975c96dc-mrtkt**
- Action: Identified readiness probe failure due to a 404 HTTP return.
- Result: ⚠️ Requires adjustment in liveness/readiness configuration.
- Before: Status - CrashLoopBackOff due to probe failure on non-existing path /nonexistent.
- After: Requires remediation to point to a valid endpoint for probes.
3. **Pod: default/image-pull-fail-d47bdcd75-sbxkw**
- Action: Identified ImagePullBackOff from non-resolvable registry URL.
- Result: ⚠️ Requires updating image source/repository.
- Before: Status - ImagePullBackOff from nonexistent URL nonexistent-registry.io/fake/image:v1.0.0.
- After: Pending update of image source in deployment configuration.
4. **Pod: makdo-test/resource-starved**
- Action: Identified Pending status due to over-requested resources.
- Result: ⚠️ Requires adjustment of resource requests.
- Before: Status - Pending with excessive CPU and Memory request.
- After: Pending review and adjustment of resource requests.
Skipped Actions:
- No direct actions are performable without application/deployment configuration modifications.
Verification:
- Pods now running: None of the identified pods require immediate running remedies as issues demand configuration changes.
- Pods still failing: Multiple as identified; awaiting operational redesign.
- Issues resolved: 0
- Issues remaining: 4
Summary:
- Total actions attempted: 4
- Successful: 0 (Identified and pending corrective configurations)
- Failed: 0
- Requires approval: 4 (Due to configuration changes)

🎉 **MAKDO Demo Complete!**

Results:

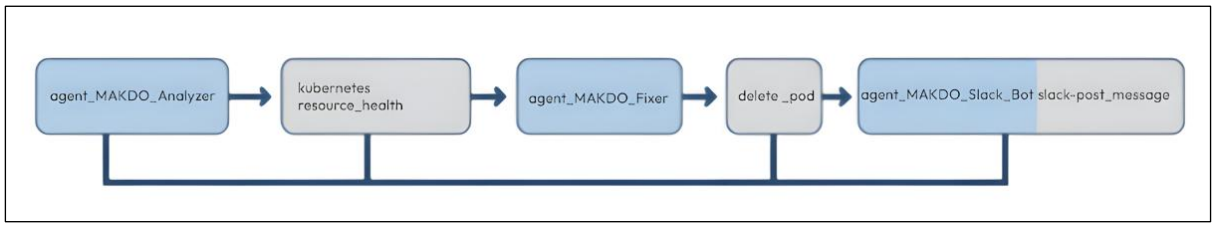
✅ Multi-agent coordination successful
✅ Analyzer detected cluster issues via k8s-ai
✅ Fixer attempted remediation
✅ Full autonomous DevOps workflow demonstrated

Architecture validated:

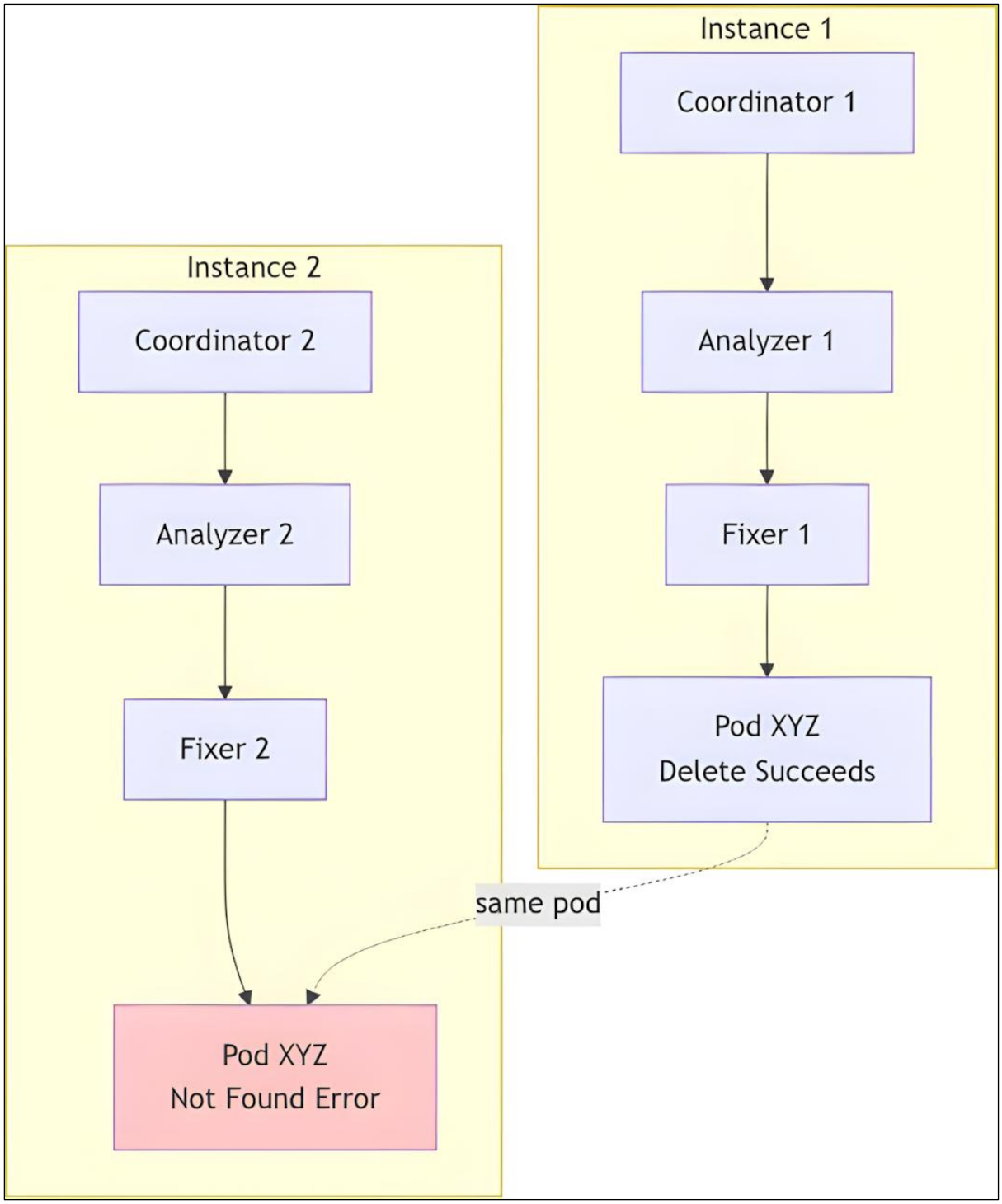
- Coordinator orchestrated all agents
- Analyzer used A2A protocol to k8s-ai
- Fixer executed safe remediation
- Slack agent provided human visibility

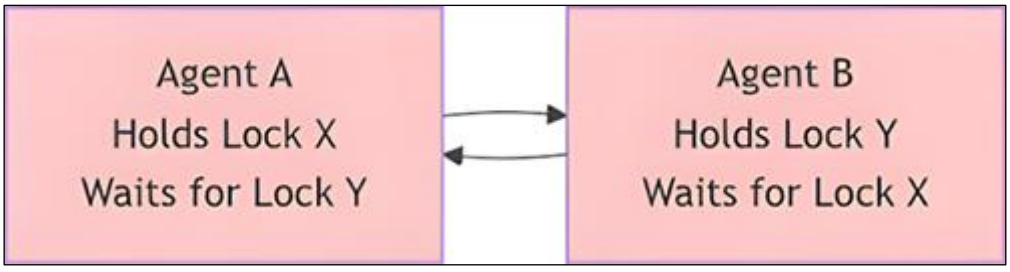
Demo completed at: 2025-10-05 21:45:21

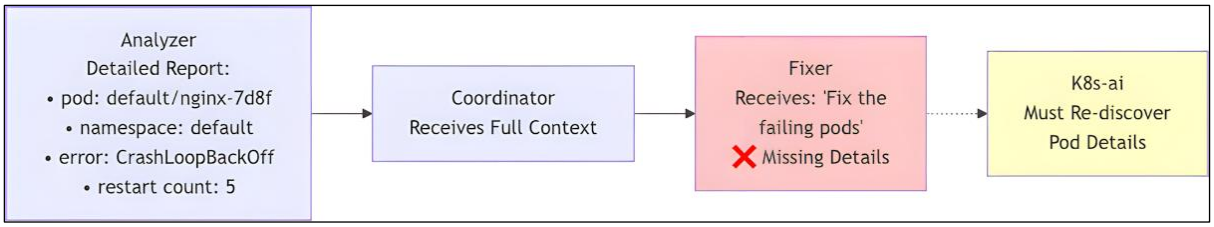
Chapter 10: Testing, Debugging, and Troubleshooting Multi-Agent Systems

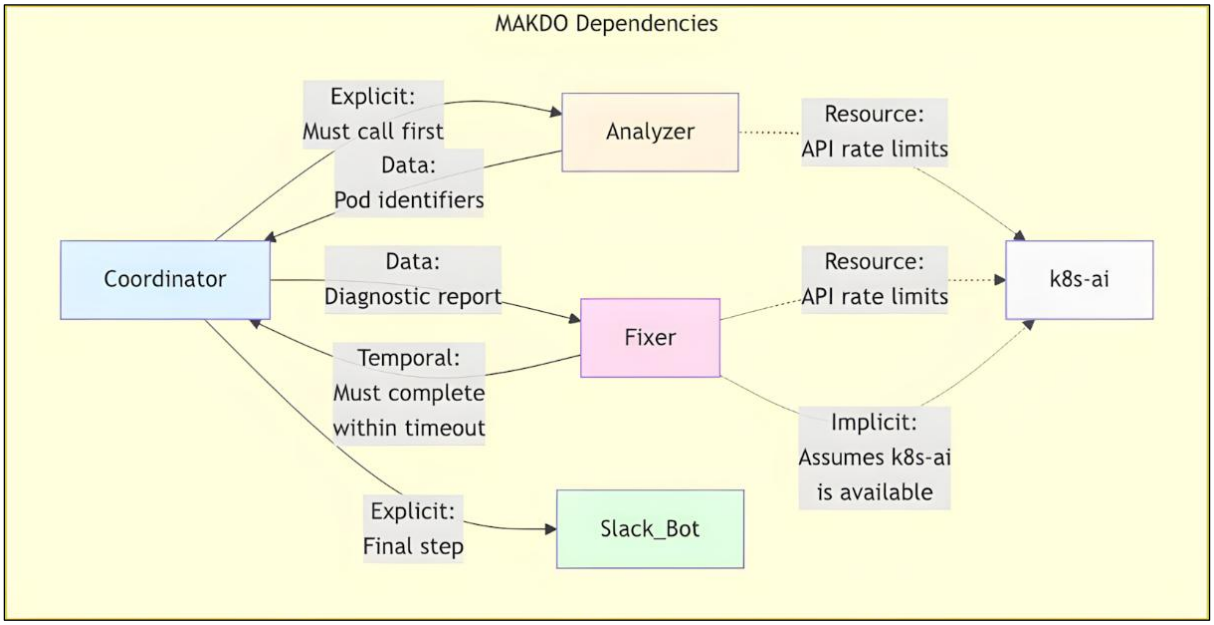


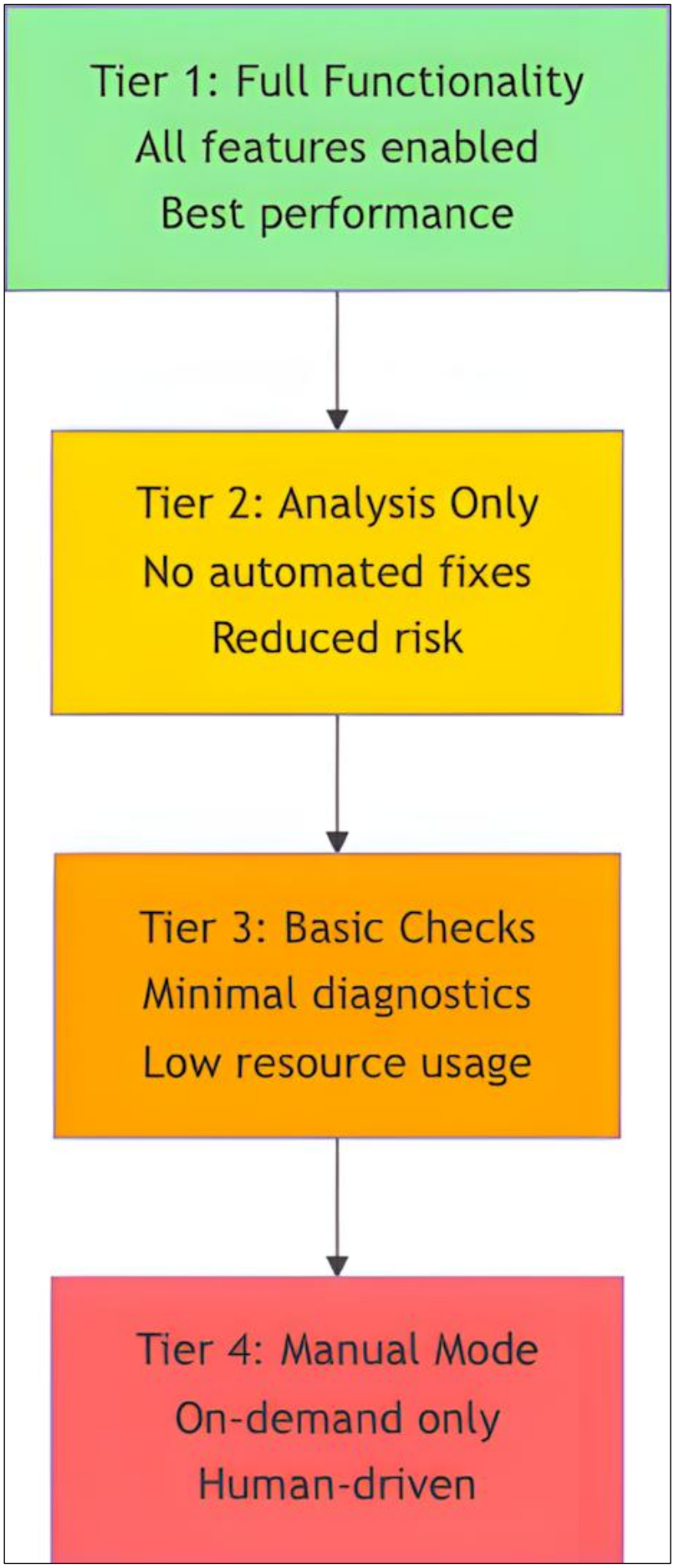


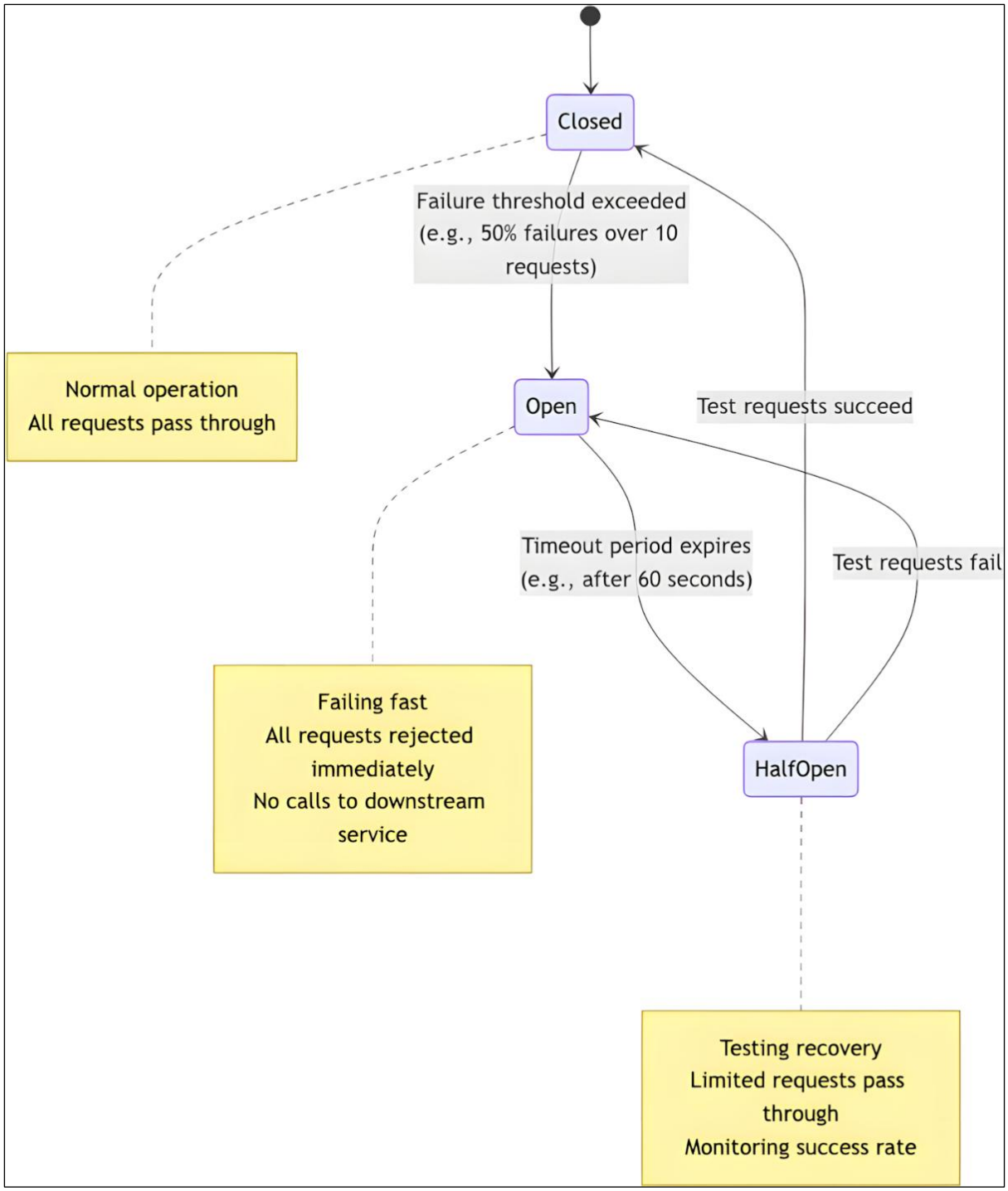

Instance 1
Coordinator 1
Analyzer 1
Fixer 1
Pod XYZ
Delete Succeeds
Instance 2
Coordinator 2
Analyzer 2
Fixer 2
same pod
Pod XYZ
Not Found Error


Agent A
Holds Lock X
Waits for Lock Y
Agent B
Holds Lock Y
Waits for Lock X


Analyzer
Detailed Report:
• pod: default/nginx-7d8f
• namespace: default
• error: CrashLoopBackOff
• restart count: 5
Coordinator
Receives Full Context
Fixer
Receives: 'Fix the failing pods'
Missing Details
K8s-ai
Must Re-discover Pod Details


MAKDO Dependencies
Explicit:
Must call first
Data:
Pod identifiers
Analyzer
Resource:
API rate limits
Coordinator
Data:
Diagnostic report
Fixer
Resource:
API rate limits
k8s-ai
Temporal:
Must complete within timeout
Implicit:
Assumes k8s-ai is available
Explicit:
Final step
Slack_Bot


Tier 1: Full Functionality
All features enabled
Best performance
Tier 2: Analysis Only
No automated fixes
Reduced risk
Tier 3: Basic Checks
Minimal diagnostics
Low resource usage
Tier 4: Manual Mode
On-demand only
Human-driven


Closed
Failure threshold exceeded
(e.g., 50% failures over 10
requests)
Normal operation
All requests pass through
Open
Test requests succeed
Timeout period expires
(e.g., after 60 seconds)
Test requests fail
Failing fast
All requests rejected
immediately
No calls to downstream
service
HalfOpen
Testing recovery
Limited requests pass
through
Monitoring success rate