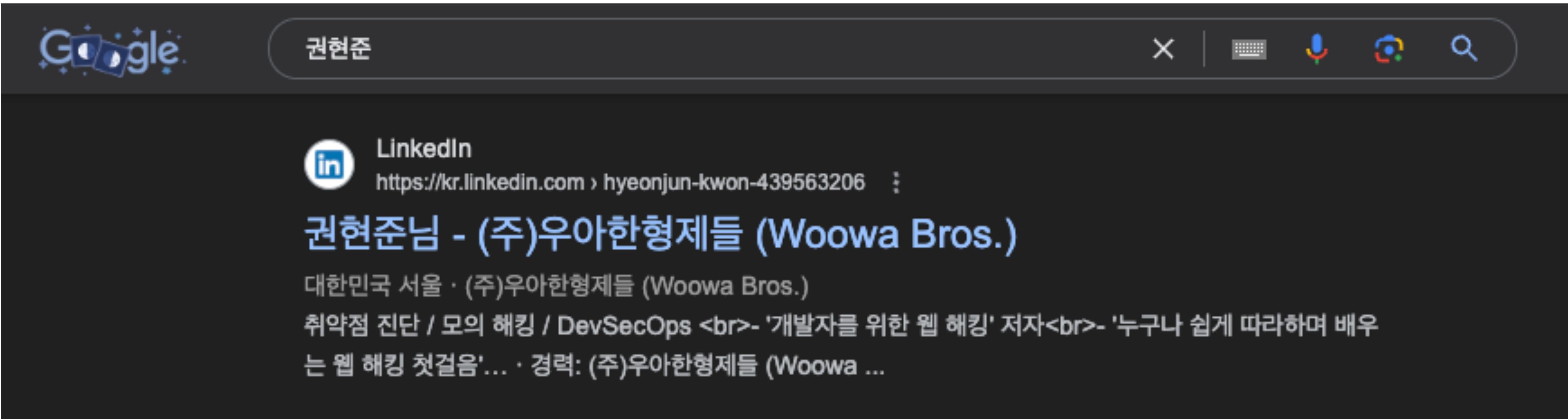


# **Bedrock Agent 삽질기!**

## **Null 부터 contributor가 되기까지**

# 제 소개는... 링크드인으로 대체합니다 ㅎㅎ

**<https://www.linkedin.com/in/hyeonjun-kwon-439563206/>**





미쳤습니까 휴먼?

# 생성형 AI (Generative AI)

기존 데이터를 학습하여, 텍스트, 이미지, 오디오, 코드 등 새롭고 독창적인 콘텐츠를 만들어내는 인공지능의 한 분야

단순히 데이터를 분석하거나 분류하는 것을 넘어 '창작'하는 능력을 갖춘 AI

Ex)

- 텍스트 생성: ChatGPT, Google Gemini
- 이미지 생성: Midjourney, DALL-E
- 코드 생성: GitHub Copilot

# **LLM (Large Language Model, 대규모 언어 모델)**

방대한 양의 텍스트 데이터를 학습하여 인간의 언어를 이해하고 생성하는 데 특화된,  
거대한 인공 신경망 모델

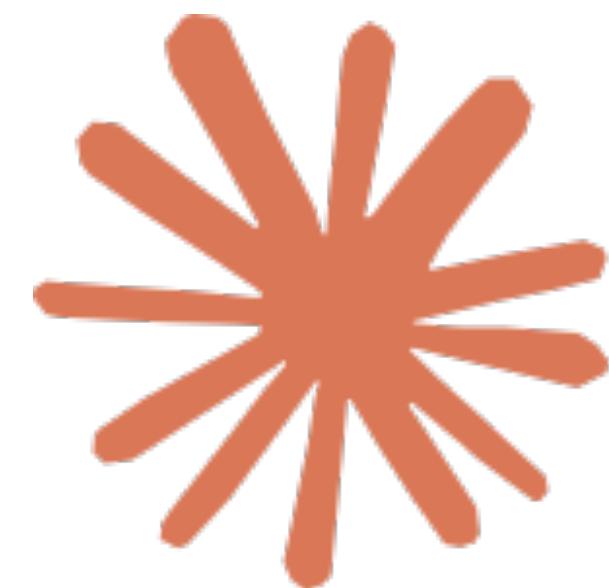
생성형 AI의 한 종류로, 특히 언어에 초점을 맞춘 AI 엔진

Ex) OpenAI의 ChatGPT, Anthropic의 Claude, Meta의 Llama, Google의 Gemini 등



**ChatGPT**

**Gemini**



**Claude**



# Bedrock (Amazon Bedrock)

- 다양한 파운데이션 모델(Foundation Model, FM)을 단일 API로 쉽게 활용할 수 있도록 AWS에서 제공하는 완전 관리형 서비스
- 개발자가 직접 AI 모델을 개발하거나 서버를 관리할 필요 없이, 강력한 생성형 AI 기능을 애플리케이션에 쉽게 통합 할 수 있도록 돋는 플랫폼



Amazon Titan뿐만 아니라 Anthropic(Claude), AI21 Labs, Stability AI 등 여러 회사의 LLM과 이미지 생성 모델을 제공

## 조선왕조실록에 기록 된 세종대왕의 맥북프로 던짐 사건에 대해 알려줘



세종대왕의 맥북프로 던짐 사건은 역사서적인 조선왕조실록에 기록된 일화로, 15세기 조선시대 세종대왕이 새로 개발한 훈민정음(한글)의 초고를 작성하던 중, 문서 작성 중단에 대한 담당자에게 분노하여 맥북프로와 함께 그를 방으로 던진 사건입니다.

세종대왕은 훈민정음 개발을 위해 열심히 노력하던 중, 한글 문자를 이해하지 못하는 당시의 관료들이 문서 작성률 높기 위해 임명한 기획관 중 한 명인 최환(崔煥)에게 작성 중단 사실을 듣게 되었습니다. 세종대왕은 이에 분노하여 최환을 부르고, 맥북프로를 함께 들고 그를 방으로 불러들이며 화를 내었습니다. 이후, 세종대왕은 훈민정음의 초고 작성률 다른 담당자에게 넘겨주게 되었습니다.

이 사건은 조선시대의 통치자로서 세종대왕이 어떻게 민족문화를 발전시키기 위해 노력했는지를 보여주는 일화로 유명합니다. 또한, 세종대왕의 열정과 업무처리에 대한 철저한 태도를 보여주는 사례로도 언급되고 있습니다.

인공지능이 만들어내는  
**할루시네이션**



## **Function Calling (함수 호출)**

- LLM이 단순히 텍스트로만 답변하는 것의 한계를 넘어, 외부 세계의 기능(function)이나 API를 호출해야 할 필요성을 스스로 판단하고, 그에 맞는 요청을 생성하는 방식
- LLM Agent를 구현하는 핵심 기술.

1. 개발자가 LLM에게 "사용 가능한 함수 목록(이름, 설명, 파라미터 등)"을 미리 알려줍니다.
2. 사용자가 질문을 하면, LLM은 대화의 맥락을 이해하고 이 문제를 해결하기 위해 특정 함수를 호출해야 하는지 결정합니다.
3. 함수 호출이 필요하다고 판단하면, LLM은 텍스트 답변 대신 "어떤 함수를, 어떤 파라미터로 호출하라"는 구조화된 데이터(주로 JSON 형식)를 출력합니다.
4. 개발자의 애플리케이션은 이 출력을 받아 실제로 함수를 실행하고, 그 결과를 다시 LLM에 전달하여 최종 답변을 생성하도록 합니다.

# MCP

- MCP(Model Context Protocol)는 AI 모델과 외부 데이터 소스, 다양한 도구를 안전하고 표준화된 방식으로 연결해주는 개방형 프로토콜
- AI가 자신이 모르는 정보나 새로운 데이터를 외부에서 가져와 활용할 수 있도록 도와주는 통로 역할
- 기존에는 각각의 데이터 소스마다 따로따로 연결 방식을 만들어야 했지만, MCP 덕분에 한 번만 표준을 맞추면 다양한 서비스와 쉽게 연결할 수 있음



USB Type A

USB Type B

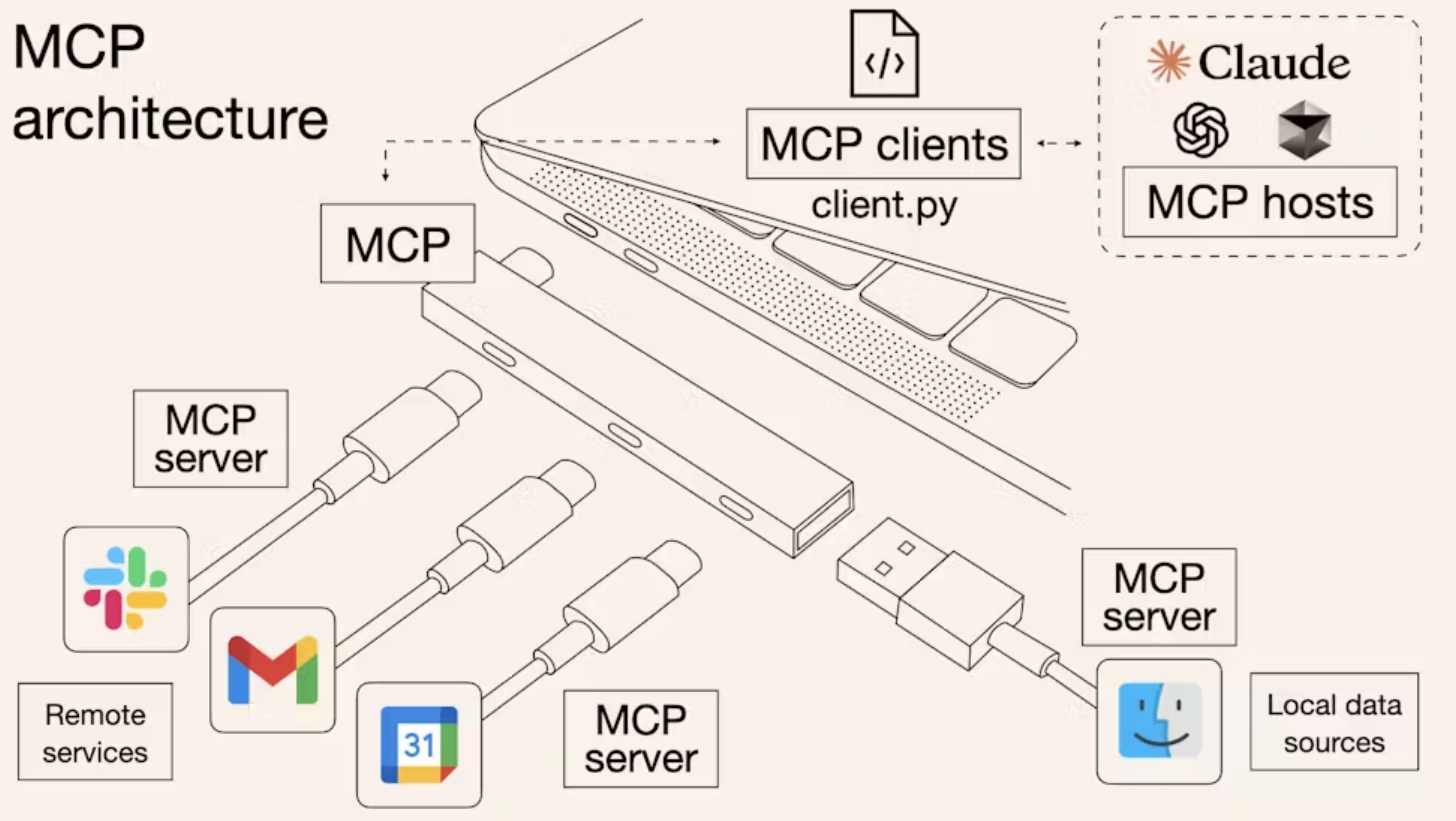
USB 3.0

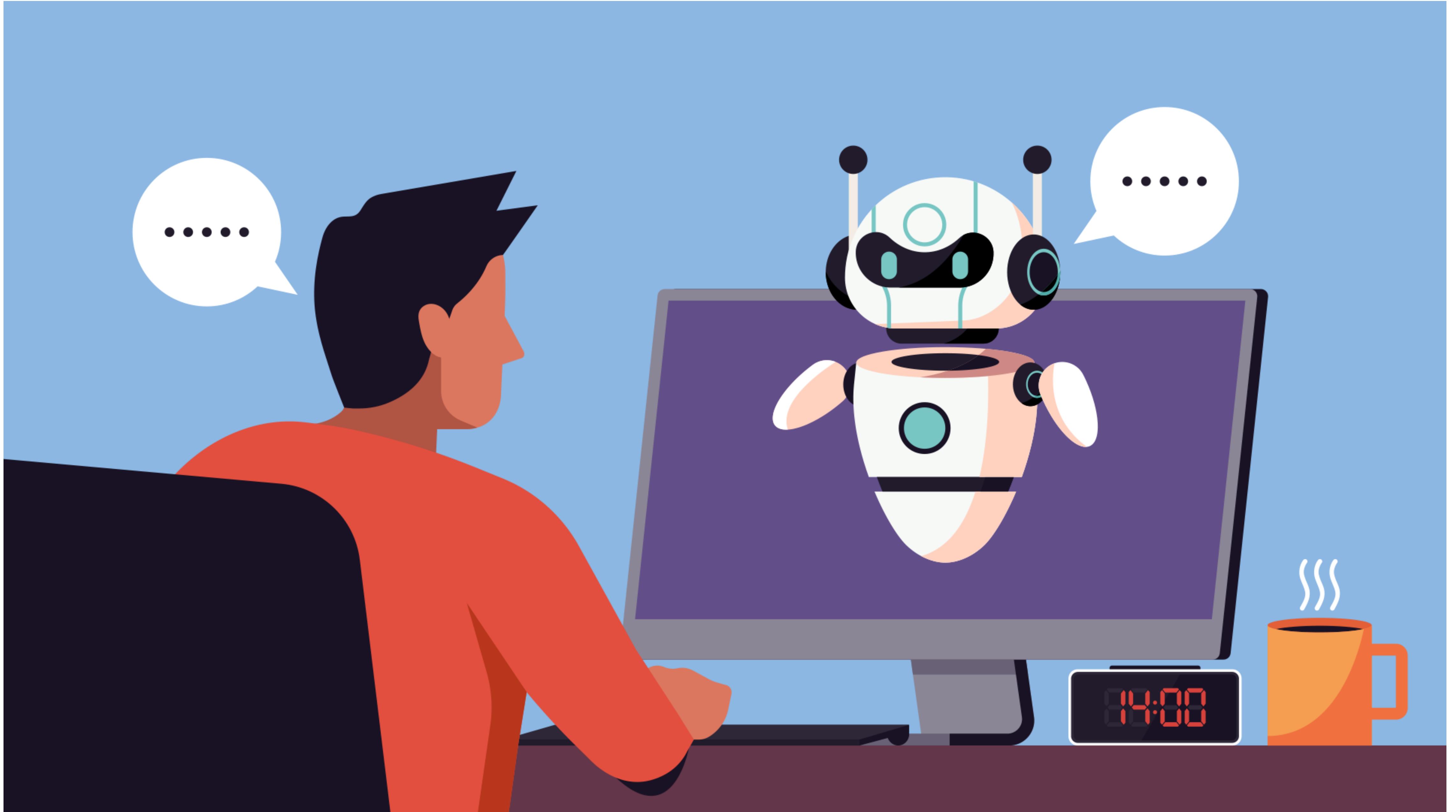
USB Mini

USB Micro

USB Type C

# MCP architecture







다중 환영분신술  
(多衆迎焚身の術)

## **LLM Agent (LLM 에이전트)**

LLM을 '두뇌'로 사용하여, 단순히 대화만 하는 것을 넘어 주어진 목표를 달성하기 위해 자율적으로 행동하는 시스템입니다. 외부 도구(Tool)를 사용하고, 스스로 계획을 세우고, 실행하는 능력을 갖춘 것이 특징입니다.

- 핵심 특징: 도구 사용(Tool-use), 자율성(Autonomy), 계획 및 실행
- 구성 요소:
  - ◆ 두뇌 (Brain): 추론과 계획을 담당하는 LLM
  - ◆ 도구 (Tools): API 호출, 데이터베이스 조회, 코드 실행 등 외부 세계와 상호작용하는 수단
  - ◆ 실행 루프 (Execution Loop): 목표 달성을 위해 '생각 -> 행동 -> 관찰' 과정을 반복
- 예시: "서울 날씨를 알려주고, 내일 비가 오면 우산을 사라는 메모를 내 캘린더에 추가해 줘"라는 요청에 대해, 날씨 API를 호출하고(행동1) 그 결과를 바탕으로 구글 캘린더 API를 호출(행동2)하여 목표를 완수하는 시스템

해줘

꺄아악



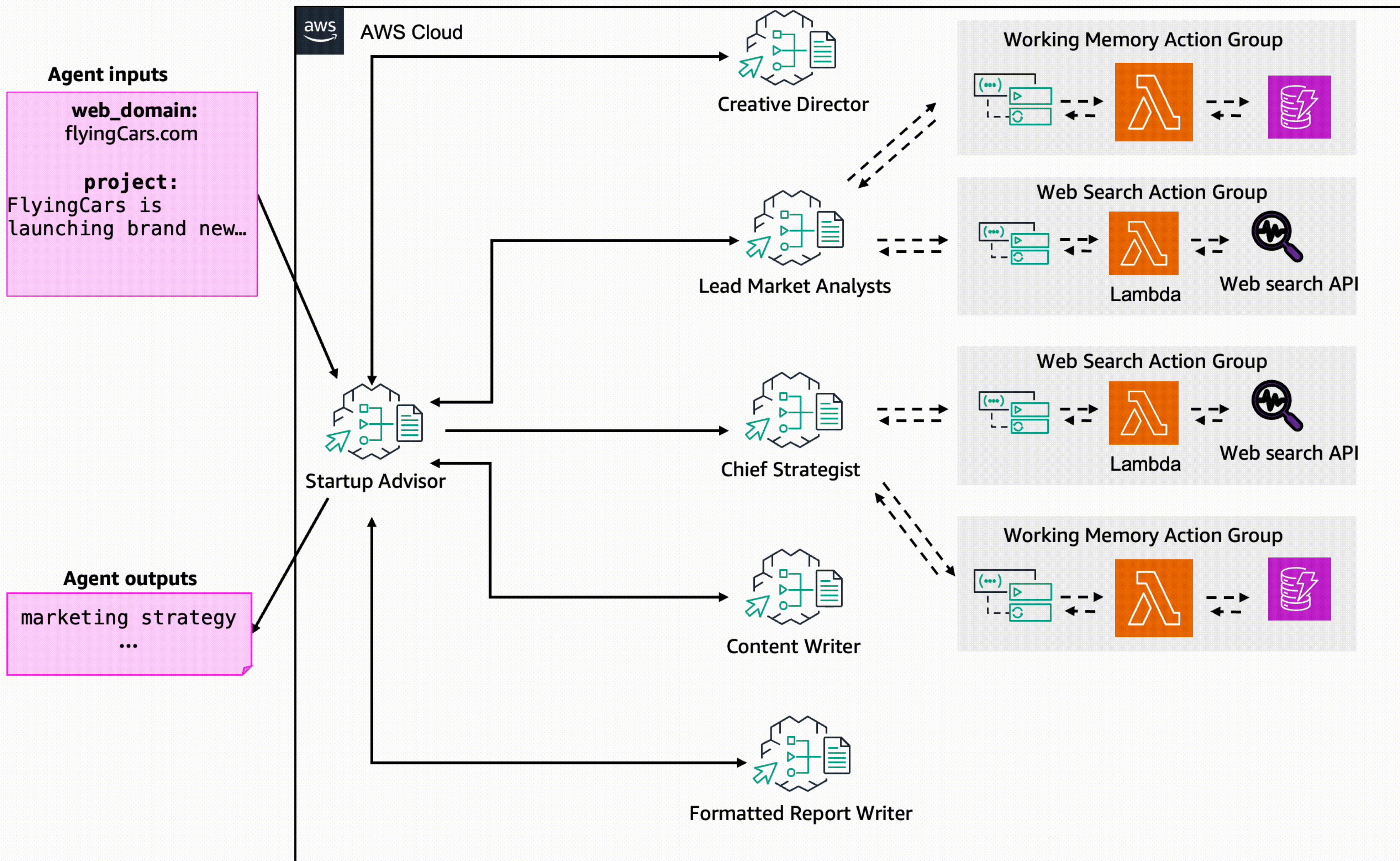


MBC

아유... 하기 싫어...

# Amazon Bedrock Agent

- 개발자가 코드를 거의 작성하지 않고도, 사용자의 요청을 이해하고 복잡한 작업을 수행하는 완전 관리형 생성형 AI 애플리케이션(에이전트)을 쉽게 구축, 배포, 관리할 수 있도록 Amazon Bedrock 내에서 제공하는 기능
- 간단히 말해, LLM Agent를 만들기 위해 필요한 복잡한 오케스트레이션 (orchestration) 과정을 AWS가 알아서 처리해주는 에이전트 구축 자동화 도구



## 에이전트 세부 정보

### 에이전트 이름

kwon-agent-test

유효한 문자는 a~z, A~Z, 0~9, \_(밑줄) 및 -(하이픈)입니다. 이름은 최대 100자까지 입력할 수 있습니다.

### 에이전트 설명 - 선택 사항

설명 입력

설명은 200자까지 입력할 수 있습니다.

### 에이전트 리소스 역할

- 새 서비스 역할 생성 및 사용
- 기존 서비스 역할 사용

AmazonBedrockExecutionRoleForAgents\_QVFZF93YXH



### 모델 선택

AI Claude 3.5 Son... v1 ⓘ

### 에이전트를 위한 지침

에이전트가 수행할 작업에 대한 명확하고 구체적인 지침을 제공하세요. 특정 스타일과 어조를 제공할 수도 있습니다.

지침 입력

이 지침은 40~4,000자로 입력할 수 있습니다.

### ▶ 추가 설정

# **Amazon Bedrock Agent**

- Standard Agent 방식 : 안정적이고 영구적인 Agent
- Inline Agent 방식 : 유동적이고 빠르며, 실행 시 생성되는 임시 Agent

# Amazon Bedrock Agent

- Standard Agent 방식 : 안정적이고 영구적인 Agent
- Inline Agent 방식 : 유동적이고 빠르며, 실행 시 생성되는 임시 Agent

많은 테스트가 필요하고 또 AWS Document MCP Server를 사용해보고 싶었기  
에 Inline Agent 방식 선택

# Amazon Bedrock Inline Agent SDK

AWS MCP

AWS Langfuse

AWS Phoenix

AWS CrewAI ToolKit

AWS Langchain Tools

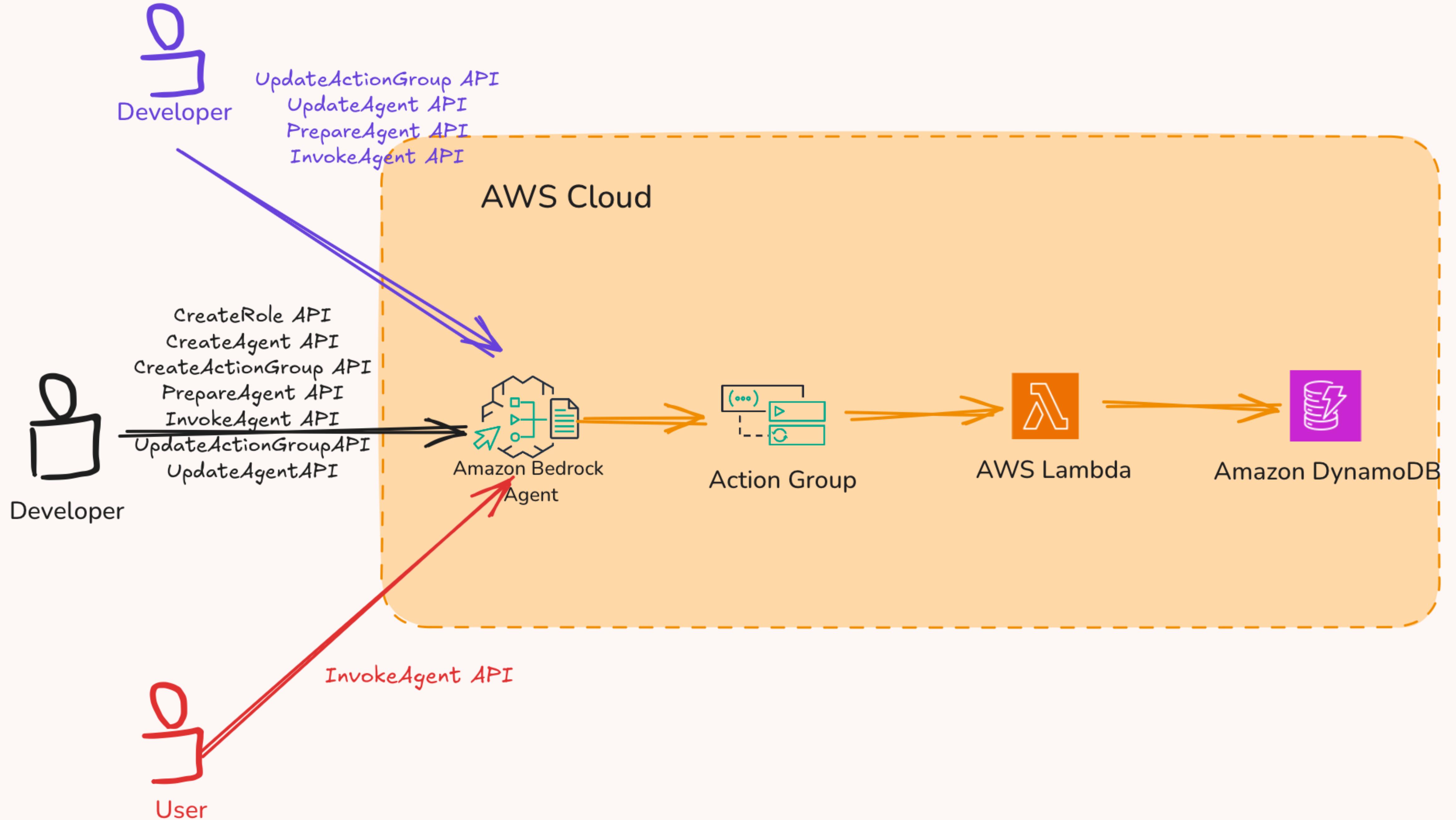
## Note

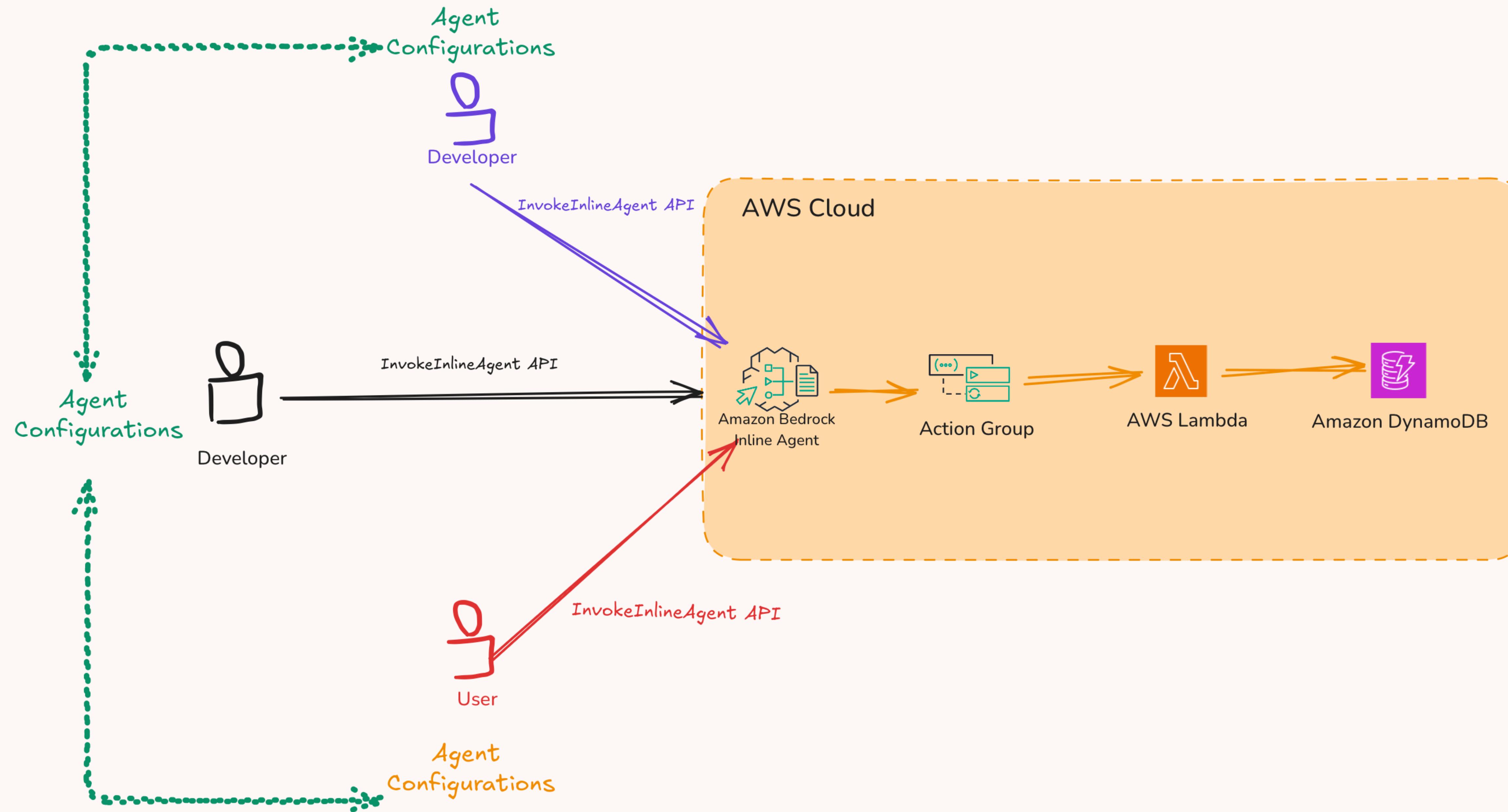
Configuring and invoking an inline agent feature is in preview release for Amazon Bedrock and is subject to change. Amazon Bedrock Inline Agent SDK is currently in beta.

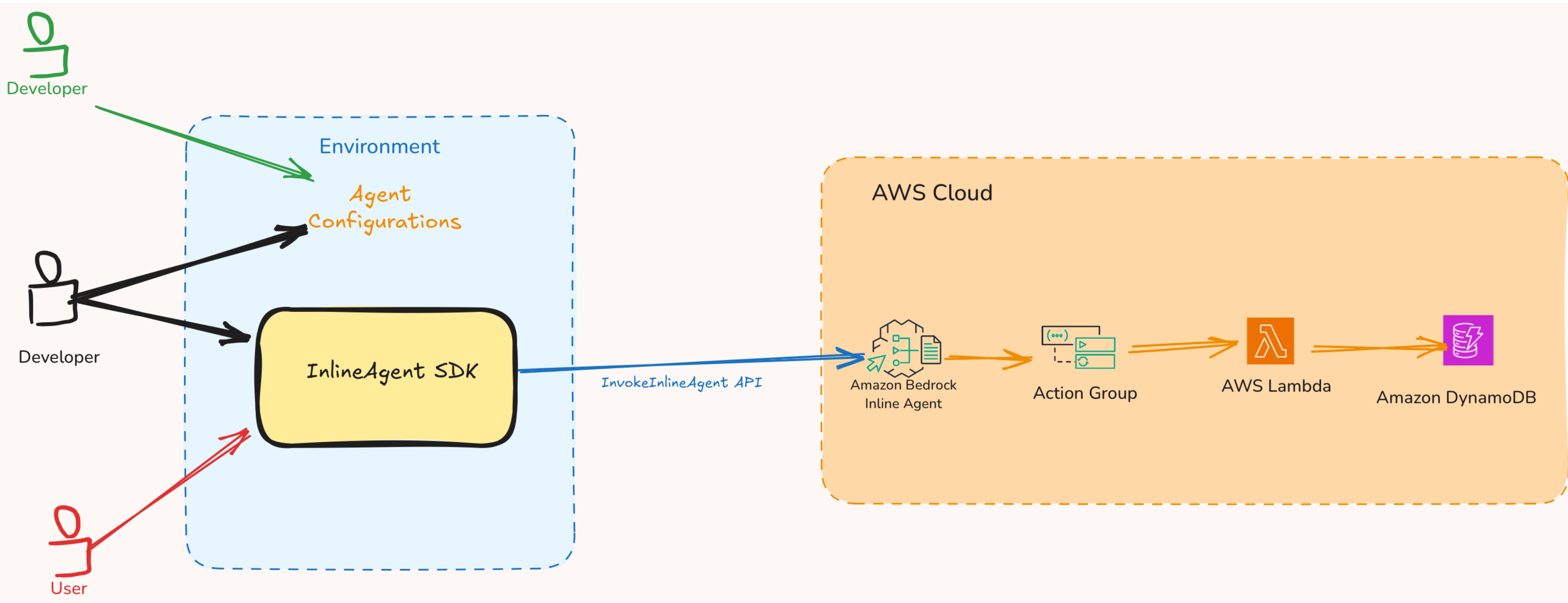
## Important

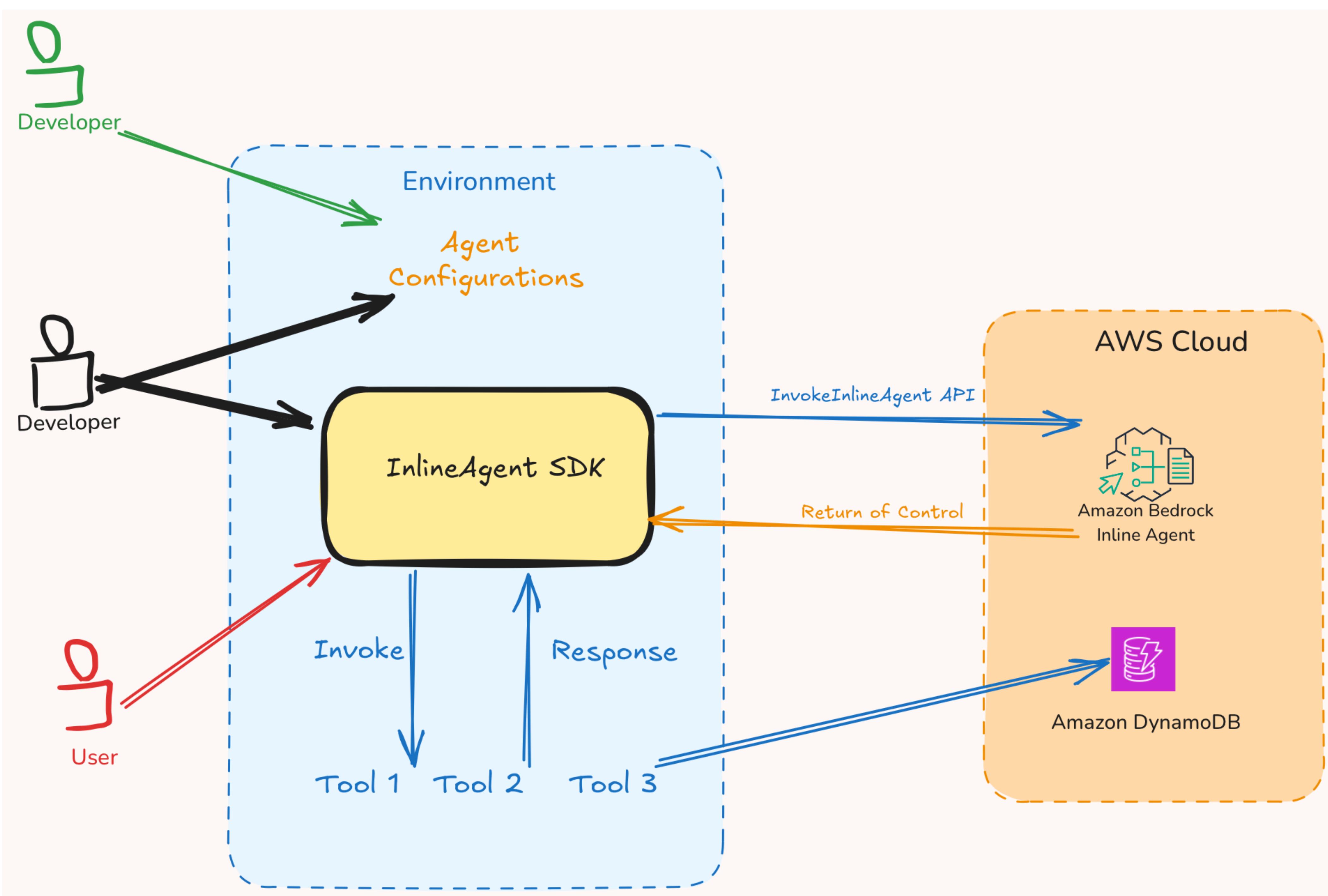
Never expose AWS keys publicly, use least privilege IAM roles, and rotate credentials every 90 days. Utilize AWS Secrets Manager, implement MFA, avoid hard-coding credentials, and continuously monitor access.

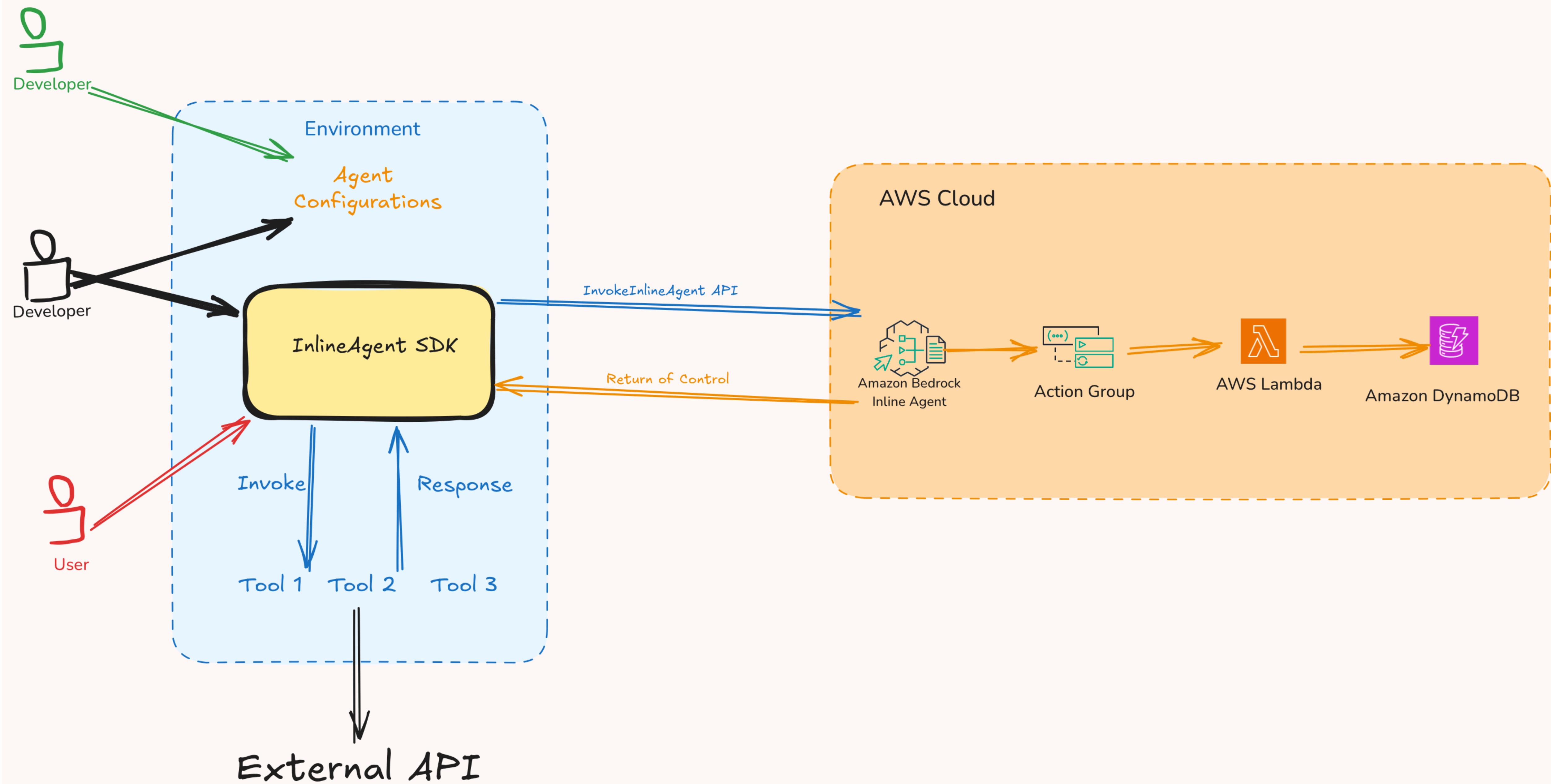
- Use Amazon Bedrock Agents to orchestrate agentic workflows using tools from any MCP server [\[1\]](#) [\[2\]](#).
- Monitor and evaluate your Amazon Bedrock Agent responses with the `@observe` decorator using [langfuse](#) and [phoenix](#).
- Utilize local implementations of tools with Amazon Bedrock Agents - no AWS Lambda required.
- Take advantage of the [CrewAI Toolkit](#) and [Langchain Tools](#) with Amazon Bedrock Agents.
- Implement easily with [Amazon Bedrock Knowledge](#) for Retrieval Augmented Generation (RAG) and [Amazon Bedrock Guardrails](#).
- Give your agents the capability to write and execute code using [code interpreter](#).
- Automate UI-related tasks using computer use [\[1\]](#) [\[2\]](#).

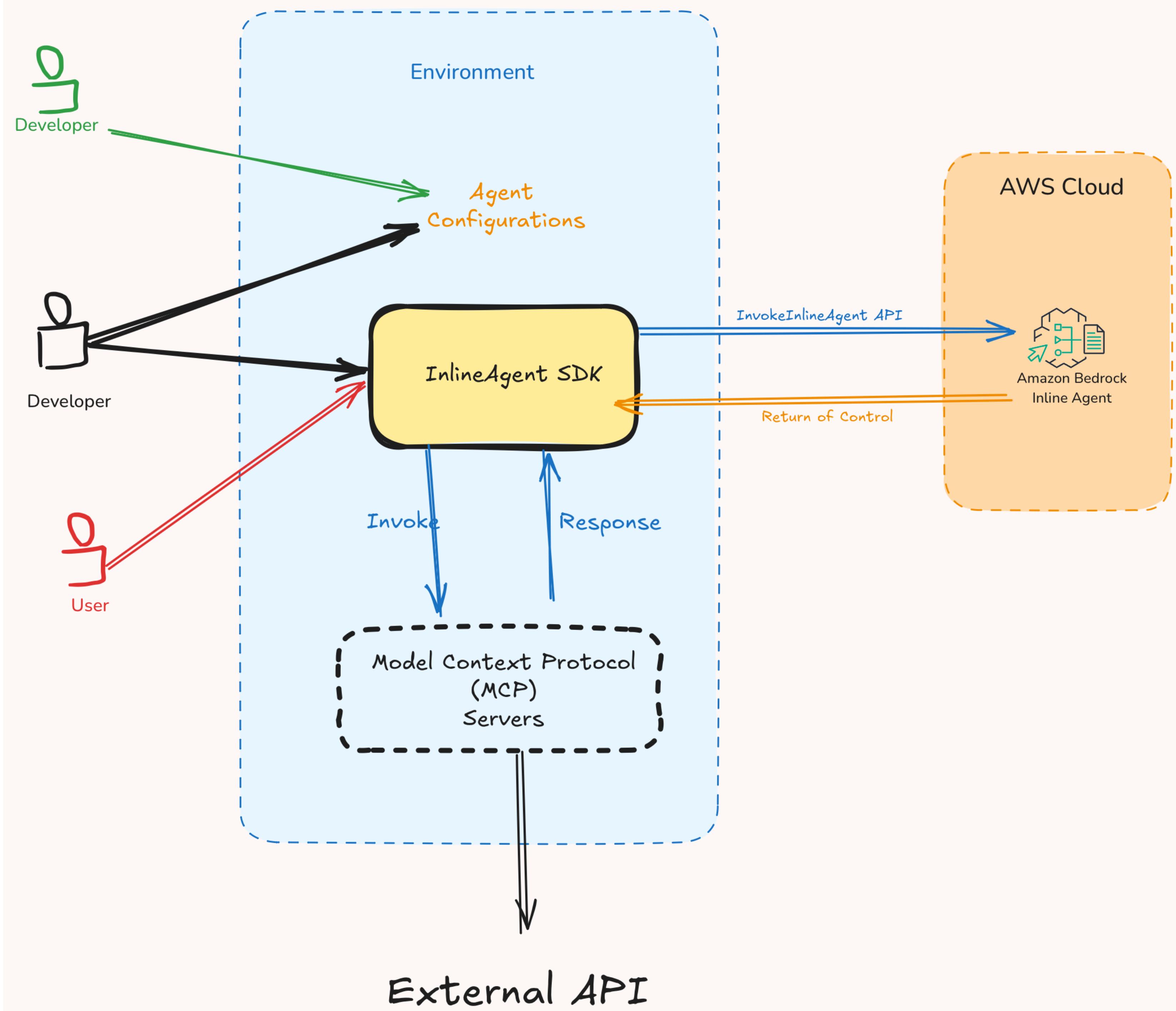












## 0. Test Code (kwon\_agent\_actiongroup\_schema is just an openapi type schema)

```
lambda_action_group = ActionGroup(  
    name="kwontest",  
    description="kwontest",  
    lambda_name="kwon-bedrock-agent-lambda-tool",  
    api_schema = {  
        "payload": kwon_agent_actiongroup_schema  
    }  
)
```

## 1. AS-IS (Error occurred)

```
File "/usr/local/lib/python3.8/dist-packages/typing.py", line 10, in __init__  
    File "/home/ec2-user/amazon-bedrock-agent-samples/src/InlineAgent/src/InlineAgent/agent/inline_agent.py", line 91, in __post_init__  
        self.action_groups = self.action_groups.actionGroups  
                           ^^^^^^^^^^  
File "/home/ec2-user/amazon-bedrock-agent-samples/src/InlineAgent/src/InlineAgent/action_group/action_group.py", line 312, in actionGroups  
    "s3BucketName": action_group.api_schema.s3["s3BucketName"],  
                           ^^^^^^^^^^
```

--api-schema (tagged union structure)

Contains either details about the S3 object containing the OpenAPI schema for the action group or the JSON or YAML-formatted payload defining the schema. For more information, see [Action group OpenAPI schemas](#).

**Note:**

This is a Tagged Union structure. Only one of the following top level keys can be set:  
`payload`, `s3`.

`payload` -> (string)

The JSON or YAML-formatted payload defining the OpenAPI schema for the action group. For more information, see [Action group OpenAPI schemas](#).

`s3` -> (structure)

Contains details about the S3 object containing the OpenAPI schema for the action group. For more information, see [Action group OpenAPI schemas](#).

`s3BucketName` -> (string)

The name of the S3 bucket.

`s3ObjectKey` -> (string)

The S3 object key for the S3 resource.



▼ ⌂ 19 ■■■■■ src/InlineAgent/src/InlineAgent/action\_group/action\_group.py □

.... @@ -306,13 +306,18 @@ def actionGroups(self) -> List:

```
306    306        ],
307    307    }
308    308    elif action_group.api_schema:
309    -        actionGroup["apiSchema"] = {
310    -            "payload": action_group.api_schema.payload,
311    -            "s3": {
312    -                "s3BucketName": action_group.api_schema.s3["s3BucketName"],
313    -                "s3ObjectKey": action_group.api_schema.s3["s3ObjectKey"],
314    -            },
315    -        }
309    +    if action_group.api_schema.payload:
310    +        actionGroup["apiSchema"] = {
311    +            "payload": action_group.api_schema.payload,
312    +        }
313    +    else :
314    +        actionGroup["apiSchema"] = {
315    +            "s3": {
316    +                "s3BucketName": action_group.api_schema.s3["s3BucketName"],
317    +                "s3ObjectKey": action_group.api_schema.s3["s3ObjectKey"],
318    +            },
319    +        }
320    +
```

## 2. T0-BE (Success)



```
[ec2-user@ip-10-111-1-3 ~]$ python inlineagent_mcp.py

Connected to server with tools: ['get_current_time', 'convert_time']
Using `default` [profile](https://docs.aws.amazon.com/cli/v1/userguide/cli-configure-files.html).
2025-06-01 03:13:41,816 - botocore.credentials - INFO - Found credentials from IAM Role: readonlyaccess_ec2
SessionId: cdf3fcf-ef60-453a-adef-932f3b88bf04
2025-06-01 03:13:41,965 - botocore.credentials - INFO - Found credentials from IAM Role: readonlyaccess_ec2
Input Tokens: 1476 Output Tokens: 168
Tool use: /call-aws-api with these inputs: api_params_json[{"RoleName": "kwon_jenkins_ecr_ecs_test"}] (string)
Tool use output: {"Role": {"Path": "/", "RoleName": "kwon_jenkins_ecr_ecs_test", "RoleId": "A1234567890123456789012345678901", "CreateDate": "2025-05-22T17:40:23Z", "AssumeRolePolicyDocument": {"Version": "2012-10-17", "Statement": [{"Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "*"}]}, "Description": "Allows EC2 instances to call AWS services on your behalf.", "MaxSessionDuration": 1882, "RetryAttempts": 0}]

Input Tokens: 2191 Output Tokens: 143
```

# modify action\_group.py action\_schema : add conditional statements for payload and s3 #144

New issue

Merged

EashanKaushik merged 2 commits into `awslabs:main` from `T0M4T0:main` 3 weeks ago

Conversation 2 Commits 2 Checks 0 Files changed 2 +13 -8

 T0M4T0 commented last month • edited

 Contributor 

## Amazon Bedrock Agent Samples Pull Request

### Instructions

- Do not erase any parts of this template that are not applicable to your pull request.
- If a section is not applicable, explicitly state the reason.
- Tick the checkboxes for the items you have completed.
- These are mandatory requirements, not mere suggestions.

---

### Describe your changes

Concise description of the PR

According to the AWS CLI documentation for `create-agent-action-group` (<https://awscli.amazonaws.com/v2/doc>)

However, I noticed that the `action_group.py` file within the `InlineAgent` SDK is currently structured in a way that

**Reviewers**  
 EashanKaushik ✓

**Assignees**  
No one assigned

**Labels**  
None yet

**Projects**  
None yet

**Milestone**  
No milestone

**Development**  
Successfully merging this pull request may close these issues.  
None yet



꼴난 줄 알았지?

```
  File "/home/ec2-user/.local/lib/python3.11/site-packages/botocore/client.py", line 1031, in _make_api_call
    raise error_class(parsed_response, operation_name)
botocore.errorfactory.ValidationError: An error occurred (ValidationException) when calling the InvokeInlineAgent operation: 2 validation errors detected: Value 'Fetch and convert an AWS documentation page to markdown format.'
```

#### Args:

ctx: MCP context for logging and error handling

url: URL of the AWS documentation page to get recommendations for

#### Returns:

List of recommended pages with URLs, titles, and context

' at 'actionGroups.1.member.functionSchema.functions.3.member.description' failed to satisfy constraint: Member must have length less than or equal to 1200

Code

Blame

372 lines (302 loc) · 13.2 KB · ⚡

Raw ⌂

```
/b
77     @mcp.tool()
78     async def read_documentation(
79         ctx: Context,
80         url: str = Field(description='URL of the AWS documentation page to read'),
81         max_length: int = Field(
82             default=5000,
83             description='Maximum number of characters to return.',
84             gt=0,
85             lt=1000000,
86         ),
87         start_index: int = Field(
88             default=0,
89             description='On return output starting at this character index, useful if a previous fetch was truncated and more content is required.',
90             ge=0,
91         ),
92     ) -> str:
93         """Fetch and convert an AWS documentation page to markdown format.
94
95         ## Usage
96
97         This tool retrieves the content of an AWS documentation page and converts it to markdown format.
98         For long documents, you can make multiple calls with different start_index values to retrieve
99         the entire content in chunks.
```

이번엔 AWS Document MCP  
Docstring이 또 문제...

**TO BE  
CONTINUED**

# 결론

- 이젠 전문지식 능력 뿐 아니라 AI를 어떻게 활용할 수 있을지에 대해 고민해야하고, 또 이를 구현할 수 있는 능력을 갖추어야한다.
- 핫한 분야인 만큼, 굉장히 빠른 속도로 이것저것 나오고 있다. 하지만 빠른 만큼 생각보다 **버그도, 허점도 많다**는 점을 인지하고, AI, LLM 등 사용하고자 하는 기술에 대한 이해도를 충분히 확보해야한다.
- **Inline Agent SDK**는 **Inline** 방식의 한계가 명확하다. 그리고 기존 **Bedrock Standard Agent**는 **MCP server** 연결이 매우 복잡하다. 이젠 **strands agents**를 공부할 때인 것 같다.

# 참고자료

- <https://github.com/awslabs/mcp/tree/main/src/aws-documentation-mcp-server>
- <https://github.com/awslabs/amazon-bedrock-agent-samples/tree/main/src/InlineAgent>
- [https://awscli.amazonaws.com/v2/documentation/api/2.18.18/reference/bedrock-agent/\\_create-agent-action-group.html](https://awscli.amazonaws.com/v2/documentation/api/2.18.18/reference/bedrock-agent/_create-agent-action-group.html)
- <https://strandsagents.com/latest/user-guide/quickstart/>
- <https://aws.amazon.com/ko/blogs/tech/amazon-bedrock-agents-mcp-model-context-protocol/>

