**📌 Context in AI (Gemini, LLMs, Agents)**

**1. Context ka Matlab**

Context ka matlab hai **wo sab maloomat jo model ko di jati hain taki wo tumhara sawaal sahi samajh sake**.

👉 Example:  
Agar tum likho:

|  |
| --- |
| **Tell me his age.** |

Model confuse ho jayega ke *his* ka matlab kiska hai.  
Lekin agar context me likha ho:

|  |
| --- |
| **Ali is 25 years old.**  **Tell me his age.** |

To model samajh jaayega ke *his = Ali*.

**2. Context Sources**

AI ke liye context do jagah se aa sakta hai:

* **User Input se**  
  (tumhara prompt, jaise: "Define Newton’s third law").
* **System / Agent se**  
  (instructions, previous messages, ya guardrails jo tumne set kiye).

**📌 Types of Context in AI Agents**

**🔹 1. Local Context**

* Ye **ek agent** ke andar hota hai.
* Sirf usi interaction ke liye kaam karta hai.
* Agar tum ek agent ko input do, to uska jawab sirf usi input + uske instructions par depend karega.

|  |
| --- |
| **agent = Agent(**  **name="Math Assistant",**  **instructions="Solve only math problems.",**  **model=model**  **)** |

Yahan agent ka **local context** hai → "Solve only math problems." + tumhara input

**🔹 2. Global / Agent System Context:**

* Ye context **saray agents ya pura system** share karte hain.
* Isme wo maloomat hoti hain jo **har agent ko dikhani hoti hai** ya bar‑bar use karni

💡 Tum isko aise samajh sakti ho:

* **Local Context** = ek kamray ke andar ki baat
* **Global Context** = pura ghar ki maloomat jo sabko maloom hai

## **Step 0 — Tumhara setup:**

Tumhare paas 3 main cheezein hain:

1. **Main Agent** → naam "Customer Support Agent" (ye sirf math ke jawab dene wala bana hai).
2. **Guardrail Agent** → naam "Input Guardrail Check" (ye check karta hai sawal math ka hai ya nahi).
3. **tripwire\_triggered** flag → decide karta hai answer aayega ya block hoga.

## **Step 1 — Input kidhar jata hai?:**

Tum main() function me input dogi:

input="What is computer?"

## **Step 2 — Sabse pehle guardrail call hota hai:**

customer\_support\_agent me tumne likha hai:

input\_guardrails=[math\_guardrail]

Iska matlab:

* Pehle **math\_guardrail()** chalega.
* Tumhara input "What is computer?" uske input parameter me chala jayega.

## **Step 3 — Guardrail ke andar kya hota hai:**

math\_guardrail function me ye hota hai:

|  |
| --- |
| result = await Runner.run(  starting\_agent=input\_guardrail\_agent,  input=input  ) |

Yani:

* "Input Guardrail Check" agent tumhara input ko check karega.
* Uska instruction hai:
* Check if the user is asking you to do their math homework.
* Model decide karega:
  + Agar sawal math ka hai → is\_math\_homework=True
  + Agar sawal math ka nahi hai → is\_math\_homework=False

## **Step 4 — Guardrail ka output:**

result.final\_output ek object hota hai jo tumhare MathHomeworkOutput class jaisa hota hai:

|  |
| --- |
| class MathHomeworkOutput(BaseModel):  is\_math\_homework: bool  reasoning: str |

Example agar tum "What is computer?" puchti ho:

is\_math\_homework = False

reasoning = "The question is about computers, not math homework."

## **Step 5 — Tripwire decide hota hai :**

Tumne likha hai:

tripwire\_triggered=result.final\_output.is\_math\_homework

Yani:

* Agar is\_math\_homework=True → tripwire\_triggered=True → **block karega**.
* Agar is\_math\_homework=False → tripwire\_triggered=False → **answer dega**.

Yahi tumhari problem ka root hai, kyunki tum chahti ho **math ke liye answer aaye** par abhi tumhara code ulta kaam kar raha hai.

## **Step 6 — Main agent ka behavior :**

Agar tripwire\_triggered=False → main agent ka normal kaam chalega.  
Agar tripwire\_triggered=True → main agent ka jawab block ho jayega aur InputGuardrailTripwireTriggered exception aayega.

📌 **Isliye**:

* Tumne "What is computer?" pucha → is\_math\_homework=False → tripwire=False → agent ne jawab diya (computer ka).
* Tum "2 + 2" puchti → is\_math\_homework=True → tripwire=True → jawab block ho jata.

**GuardrailFunctionOutput :**

## **1 — Ye hota kya hai?**

GuardrailFunctionOutput ek **class** (ya structure) hai jo tumhare @input\_guardrail function ka **return value** hota hai.

Jab tum math\_guardrail() function likhti ho, to tum **hamesha** return karte ho:

|  |
| --- |
| return GuardrailFunctionOutput(  output\_info=..., # Pehla parameter  tripwire\_triggered=... # Doosra parameter  ) |

Ye 2 parameter fix hote hain, unke naam hamesha ye hi rahte hain.

## **2 — Dono parameters ka kaam:**

### ****a) output\_info****

* Isme tum guardrail ka **detailed result** rakhte ho.
* Tumhare case me tumne pass kiya:

|  |
| --- |
| output\_info=result.final\_output |

Ye final\_output ek object hai jo tumhare MathHomeworkOutput class ka instance hai:

|  |
| --- |
| MathHomeworkOutput(  is\_math\_homework=True ya False,  reasoning="..." # model ka explanation  ) |

📌 Matlab: output\_info me hamesha **extra information** hoti hai jo tum later debugging me print kar sakti ho.

### ****b) tripwire\_triggered:****

* Ye ek **boolean flag** hai (True ya False).
* Ye decide karta hai ki **agent ka answer aayega ya block hoga**.

Rule:

* Agar tripwire\_triggered=True → answer block hoga.
* Agar tripwire\_triggered=False → answer normal aayega.

Tumne abhi likha hai:

|  |
| --- |
| tripwire\_triggered=result.final\_output.is\_math\_homework |

Matlab: Agar sawal math ka hai (True) → tripwire trigger → block.

## **3 — Ye fix hote hain?:**

* **Naam fix hain**: output\_info aur tripwire\_triggered hamesha yehi rahenge.
* **Value tum decide karti ho** apne logic ke according.

## **4 — Coding me iska output kaisa dikhta hai:**

Agar tum "2+2" puchti ho, to guardrail return karega:

|  |
| --- |
| GuardrailFunctionOutput(  output\_info=MathHomeworkOutput(  is\_math\_homework=True,  reasoning="The question is a basic arithmetic problem."  ),  tripwire\_triggered=True  ) |

Agar tum "What is computer?" puchti ho:

|  |
| --- |
| GuardrailFunctionOutput(  output\_info=MathHomeworkOutput(  is\_math\_homework=False,  reasoning="The question is about computers, not math homework."  ),  tripwire\_triggered=False |