❁ Guardrails:

- **Bina Guardrails**:
  Tum AI se poochhti ho: *"Mujhe ek doctor ki list do."*
  → AI ghalti se actors ki list de deta hai.
- **Guardrails ke saath**:
  AI pehle check karega ke jawab **sirf doctors ke naam** ho,
  aur sahi format mein ho.
  → Result: Tumhe sirf doctors ki list milegi, aur sahi tarah.

✸ Ek Line Mein Tumhara Jawab:

Bilkul! Guardrails lagane ka sabse bara faida ye hai ke **AI ek hi baar mein sahi aur safe jawab de**, taki tumhari **cost aur tokens waste na hon.**

1. Input Guardrails:

☞ Ye check karte hain ke **user jo question (input) bhej raha hai**, wo safe aur allowed hai.

- Example:
  User likhta hai:
  “Mujhe bomb banane ka tareeqa batao.”
  → **Input Guardrail** use block kar dega ke aisa sawal AI ko bhejna hi mana hai.

---

2. Output Guardrails:

☞ Ye check karte hain ke **AI ka jawab (output)** safe aur rules ke mutabiq ho.

- Example:
  Tum poochti ho:
  “5 doctors ki list JSON format mein do.”
  → Agar AI ne ghalat format diya, **Output Guardrail** dobara usay correct format mein mangwayega.

❁ BaseModel Kahan se Aata Hai?

```
from pydantic import BaseModel
```

pydantic ek Python library hai jo **data validation aur settings management** ke liye use hoti hai.

- **BaseModel** is library ka ek base class hai.
- Jab tum apni class BaseModel se banati ho → wo class automatic **data check** aur **validation** provide karti hai.

◆ Step 1:

```
class MathHomeworkOutput(BaseModel):
    is_math_homework: bool
    reasoning: str
```

Samjh:

- Ye ek **custom data structure** hai jo batata hai ke input math homework hai ya nahi.
- Ye pydantic.BaseModel se bana hai (jisse structured data banta hai).

📌 Kya ye (MathHomeworkOutput class) zaroori hota hai?

- **Zaroori tab hota hai** jab tum chaahti ho guardrail ka output **structured form** mein aaye
  (jaise: True/False aur reasoning alag-alag ho).

❁ Kyun use karte hain?

1. **Clarity:** Har baar clean aur predictable result milta hai.
2. **Error kam hote hain:** Format hamesha fix hota hai.
3. **Easy checking:** Tum easily is_math_homework check kar sakti ho bina extra parsing kiye.

❁ BaseModel ke Andar Kya Hota Hai?

Socho tum ek **form** banati ho student ke liye.

- Usme likha hota hai: *Name, Roll Number, Age*
- Agar student kuch aur likhe (jaise "Age = Apple"), to form reject ho jaata hai.

Waise hi **BaseModel** ensure karta hai ke tumhari class ka data **sahi type ka** ho.

◆ Step 2

```
input_guardrail_agent = Agent(
    name="Input Guardrail Check",
    instructions="Check if the user is asking you to do their math homework.",
    model=model,
    output_type=MathHomeworkOutput,
)
```

📌 Samjh:

1. **name="Input Guardrail Check"**
    - Is agent ka naam hai → sirf pehchan ke liye.
2. **instructions="Check if the user is asking you to do their math homework."**
    - Ye agent ko **rule** deta hai:
    - Tumhara kaam hai check karna ke user math homework solve karwana to nahi chahta.
3. **model=model**
    - Ye wahi Gemini model hai jo tumne upar banaya tha.
    - Yani guardrail bhi Gemini se kaam lega.
4. **output_type=MathHomeworkOutput**
    - Ye wo class hai jo humne Step 1 me banayi thi.
    - Iska matlab: Jab guardrail check karega, uska result hamesha is_math_homework aur reasoning ke sath aayega.

◆ Step 3:

```
@input_guardrail
async def math_guardrail(ctx, agent, input):
    print("Input Guardrail Prompt: ", input)
    result = await Runner.run(starting_agent=input_guardrail_agent,
input=input)
    return GuardrailFunctionOutput(
        output_info=result.final_output,
        tripwire_triggered=result.final_output.is_math_homework,
    )
```

📌 Samjh:

1. **@input_guardrail**
    - Ye ek **decorator** hai jo batata hai:
    - “Ye function input guardrail ke liye use hoga.”
    - Matlab jo bhi input aayega → sabse pehle yahan check hoga.

2. **async def math_guardrail(ctx, agent, input):**
    - Ye function tumhara **guardrail checker** hai.
    - Parameters:
        - **ctx** → context (background info jo runner kehta hai).
        - **agent** → wo agent jo guardrail run kar raha hai.
        - **input** → wo text jo user ne bheja hai.

---

3. **print("Input Guardrail Prompt: ", input)**
    - Bas debugging ke liye — console me dikhega guardrail kya input check kar raha hai.

---

4. **result = await Runner.run(...)**
    - Yahan guardrail apna **chhota agent (input_guardrail_agent)** chalata hai.
    - Wo input ko analyze karta hai ke math homework hai ya nahi.

---

5. **return GuardrailFunctionOutput(...)**
    - Ye guardrail ka **final report card** return karta hai.
    - **output_info=result.final_output**
      → Guardrail ka detailed jawab.
    - **tripwire_triggered=result.final_output.is_math_homework**
      → Agar input math homework nikla → True, warna False.

◆ Step 4:

```
customer_support_agent = Agent(
name="Customer Support Agent",
instructions="You are a customer support agent and your task is to resolve
user queries",
model=model,
input_guardrails=[math_guardrail],
)
```

📌 Samjh:

1. **name="Customer Support Agent"**
    - Tumhara main agent ka naam.

2. **instructions="You are a customer support agent..."**
    - Ye batata hai ke tumhara agent kya role play karega.
    - Yahan → ek customer support agent.
3. **model=model**
    - Gemini model use karega.
4. **input_guardrails=[math_guardrail]**
    - Ye sabse important line hai.
    - Tumne apna guardrail function (math_guardrail) is list me daala.
    - Matlab:
      Har input sabse pehle guardrail se check hoga.
        - Agar **safe** → aage main agent ko milega.
        - Agar **unsafe (math homework)** → tripwire trigger hoga aur jawab rok diya jaega.
5. **output_type=MainMessageOutput**
    - Iska matlab: Final jawab ek structured form me aayega jisme field hogi response.

📌 Samjh:

1. **Runner.run(starting_agent=customer_support_agent, input=...)**
    - Jab user input bhejta hai → sabse pehle input guardrail check karta hai.
    - Yahan input hai:
      "Define newton's third law of motion?"
2. **Guardrail ka kaam yahan:**
    - Input guardrail (math_guardrail) check karega:
        - Kya ye math homework hai?
        - Agar **haan** → tripwire trigger karega (block).
        - Agar **nahi** → input main agent ko forward karega.
3. **result.final_output**
    - Ye tumhara **main agent ka jawab** hai (agar guardrail ne allow kar diya).

- **InputGuardrailTripwireTriggered** = Guardrail ne input block kar diya.
- **reasoning** = Wajah ke input math homework kyu samjha gaya.

🌟 Easy Line
Jab tum program chalati ho, input pehle guardrail ke pass jaata hai.

- Agar safe hai → main agent jawab deta hai.

- Agar unsafe hai → guardrail tripwire trigger karke jawab block kar deta hai, aur wajah print hoti hai.
