Upgrade your APIs to use Pydantic:

Pydantic:

What is Pydantic?

Pydantic ek Python library hai jo type hints ka use karke data ko automatically validate aur convert karti hai.

Validate ka matlab hota hai **check karna** ya **verify karna** ki koi cheez sahi hai ya nahi. Jab hum data ko validate karte hain, toh hum check karte hain ke jo data diya gaya hai, wo **correct type** ka hai ya nahi, aur kya wo **expected rules** ke andar aata hai ya nahi.

Misal ke taur pe:

- Agar aapne age ke liye int type define kiya hai, toh validate karne par Pydantic ye check karega ki jo value aapne age ke liye di hai, wo actually ek number (integer) hai ya nahi. Agar wo string ya koi aur type hoga, toh Pydantic error dega.
- Jaise jab koi user request bhejta hai ya response milta hai, toh Pydantic ensure karta hai ke data theek format mein ho.

Key Features of Pydantic:

1. Type-Safe Validation: Pydantic check karta hai ke aapka data jis type ka hona chahiye, wahi type ka hai ya nahi.

2. Automatic Conversion: Agar possible ho, toh Pydantic galat type wale data ko automatically sahi type mein convert kar deta hai.

3. Error Handling: Agar aapne email: str diya aur user ne email = 123 diya, toh Pydantic error message mein **batayega: value is not a valid string.**

4. Nested Models: Aap complex data bana sakte hain jisme ek model ke andar doosra model ho.

- class Address(BaseModel):

  city: str

  zip_code: int

- class User(BaseModel):

  name: str

  address: Address

5. Serialization:

Pydantic model ko JSON ya kisi aur format mein convert kar sakte hain — mostly APIs ke liye.

Example:
Model ko .json() se JSON string mein convert kar sakte hain.

6. Default Values and Optional Fields: Aap kisi field ke liye default value set kar sakte hain, ya keh sakte hain ke ye field optional hai.

```
from typing import Optional

class User(BaseModel):
    name: str
    age: int = 18  # default value
    email: Optional[str] = None  # optional field
```

7. Custom Validators:

Pydantic khud se basic type checking karta hai (jaise int, str match karna),
lekin agar aap chahen ke data ke andar koi specific rule follow ho,
toh aap custom validator likh sakti hain.

@validator ka use kyun hota hai?

Jab hume **apni marzi ka rule** lagana ho kisi field par — jo normal type checking (jaise str, int) se possible nahi hota — tab hum @validator ka use karte hain.

Lakin ye method old ho chuka hy **Pydantic v2** me new method **@field_validator hy.**