Chapitre 6 — Modèles Pydantic (validation & conversion)

---

1. C'est quoi ?

Pydantic est un module Python utilisé avec FastAPI pour **valider, convertir et documenter les données** entrantes et sortantes de ton API.

---

2. Pourquoi on l'utilise ?

- Pour **contrôler** que les données reçues sont du bon type (et les convertir si besoin)
- Pour **auto-générer la doc** Swagger des endpoints
- Pour **définir clairement la structure** de ce qu'attend ou renvoie une route

---

3. Vocabulaire

Terme	Rôle
`BaseModel`	classe de base Pydantic pour définir un modèle
`Field(...)`	sert à configurer chaque champ (default, alias, description...)
`validator`	décorateur pour ajouter une règle personnalisée de validation

---

4. Installation

```
pip install pydantic
```

(Facultatif si FastAPI est déjà installé, car inclus)

---

5. Exemple simple

```python
from pydantic import BaseModel

class Machine(BaseModel):
    name: str
    sector: str
    serial_number: str
    year_of_acquisition: int
    in_service: bool
    comment: str | None = None
```

---

6. Usage dans une route

```
@app.post("/machines")
def add_machine(machine: Machine):
    # ici machine est déjà un objet Python validé
    return machine.dict()
```

---

7. Bonus : héritage pour simplifier les inputs

```
class MachineIn(Machine):
    pass  # hérite des mêmes champs, mais on peut retirer ou ajuster si
besoin
```

✅Résultat :

Tu maîtrises les **modèles Pydantic** : structure des données claire, validation intégrée, auto-doc Swagger, et code plus sûr.