OOP

בנה את המחלקה שלפניך + **צור תרשים זהה ב draw io**

⏰ OOP Exercise: Class `Timer`

🤓 Topics Practiced

- Constructor: `__init__`

- String representation: `__str__`

- Developer-friendly output: `__repr__`

- Destructor behavior: `__del__`

- Dynamic attributes

📊 UML Diagram (Text-based)

```
+----------------------+
|        Timer         |
+----------------------+
| - name: str          |
| - duration: int      |
| - started_at: datetime  |
+----------------------+
| + __init__(...)      |
| + __str__() -> str   |
| + __repr__() -> str  |
| + __del__()          |
+----------------------+
```

🔢 Example Data

Create two timer instances:

- `Timer("Workout", 45)`

- `Timer("Meditation", 10)`

❓ Tasks

1. Create a class `Timer` with the attributes:

    - `name` (e.g., "Workout")

    - `duration` (in minutes)

    - `started_at` (store `datetime.now()` at creation)

2. Implement the `__init__` method to initialize all fields.

3. Implement the `__str__` method to return a friendly message like:

    `Timer 'Workout' set for 45 minutes (started at: 15:31:02)`

4. Implement the `__repr__` method to return Python code that can recreate the timer.

5. Implement the `__del__` method to print a goodbye message with timer name.

6. Add an attribute dynamically (e.g., `paused = False`) and then remove it.

7. Store all timers in a list and print the list. Ensure `__repr__` is used.

🚀 Bonus Challenge

- Use `time.sleep(1)` to simulate the timer running.

- Calculate and print how many seconds passed since the timer started.

- Clone a timer using `eval(repr(t))` and show that the clone has identical values.

📅 Reminder

Use `from datetime import datetime` at the top of your file.

בהצלחה !