

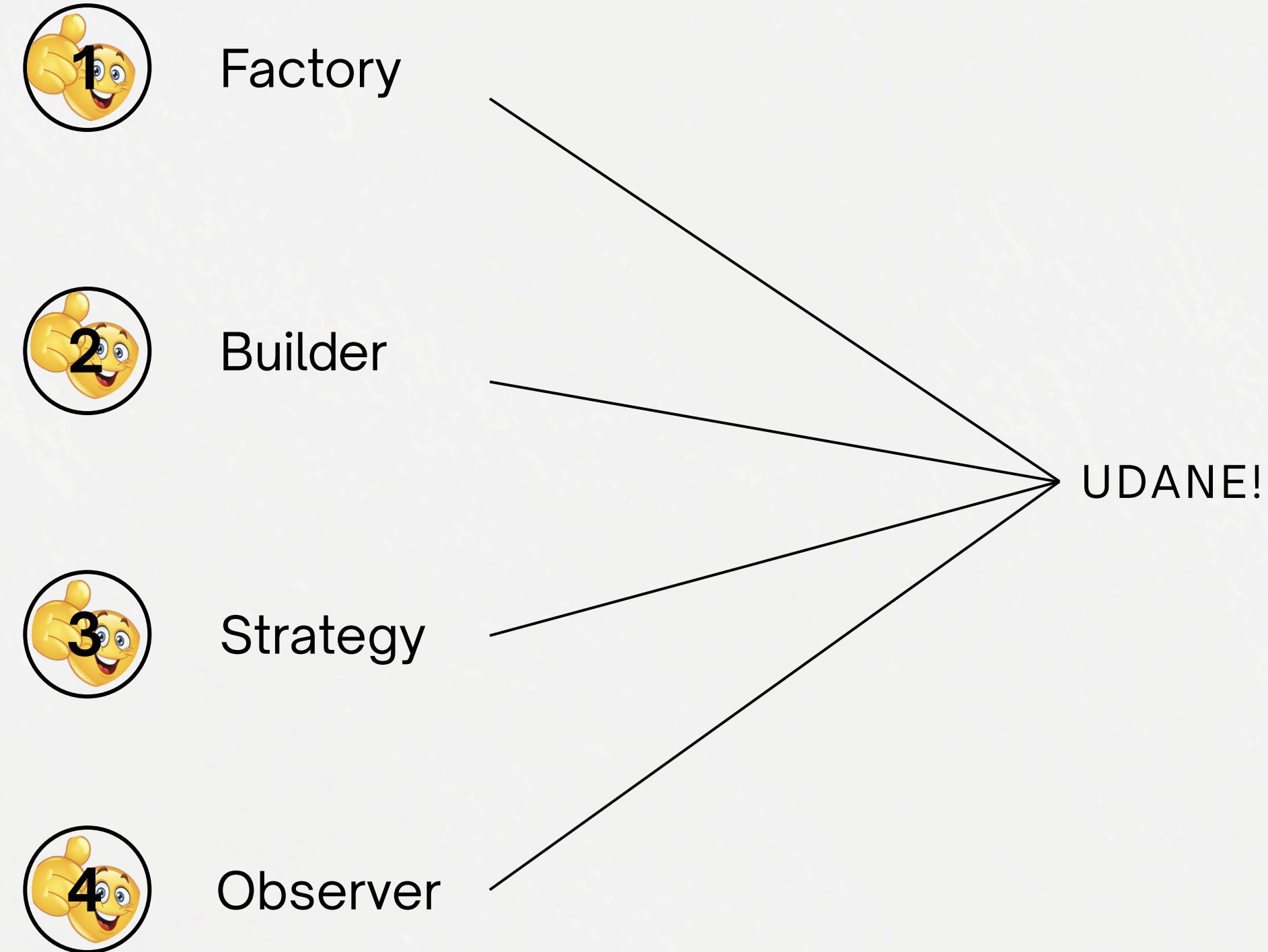
# GymUJ

PREZENTACJA KOŃCOWA + DEMO

Emilia Biros, Filip Jarzyna,  
Jakub Steć



# Planowane patterny



# “Lessons Learned”

## Separacja odpowiedzialności eliminuje God Object i chaos

Początkowy projekt był monolitem z logiką biznesową w kontrolerze i komponentach UI. Poprzez wydzielenie dedykowanych serwisów, mamy testowalny, łatwy w utrzymaniu kod.

## Abstrakcje (interfejsy) to klucz do skalowalności

Wzorce nie komplikują kodu – upraszczają go poprzez wymuszenie czystych interfejsów i jawnej struktury. Jedynie czasem wymaga od nas większej ilości kodu.

## UML to plan, nie kontrakt – realna implementacja zawsze ewoluje

Ważniejsze od sztywnego trzymania się diagramów jest dokumentowanie zmian i zachowanie spójności architektury.

# UML vs Finalny diagram klas

Usunięte klasy:

- GroupClass → zastąpione przez TrainingSession
- PersonalTraining → zastąpione przez TrainingSession
- GymEntry → przekształcone w GymServicePackage
- Trainer → dane trenera jako string w TrainingSession
- Client → zastąpione przez UserRole enum
- TimeSlot → StartTime + duration (TrainingSession)
- TicketGenerator (interfejs) → zastąpione przez TicketBuilder (builder pattern)

Zmiany:

- ServiceFactory → zmienione na SessionFactory (węższa odpowiedzialność)
- TrainingSchedule → zastąpione przez SessionManagementService
- UserSession → zastąpione przez UserSessionService

(ze względu na zmianę scope'a, uproszczenia implementacji i ograniczeń przez obecnego UMLa)

# Koniec

:)

DEMO (CLICK!).