**IDEE PATTERN – PROG3**

**Singleton**: DB

**Factory Method**: …

**Abstract Factory**: …

**Factory Pattern**: ci sono delle classi in comune con più “categorie” di persone. In base alla categoria (es. ID) viene restituito una variante di quell’oggetto (es. lista noleggio).

**Builder**: (Forse per l’HUB)/ Differenti tipi di login.

**Prototype**: …? Stessi dati.

**CoR**: noleggio e credito. (GUI)

**Command**: …

**Iterator**: per le liste in generale.

**Mediator**: comunicazione tra employee e operator.

**Memento**: per tornare indietro sul credito.

**Observer**: quando si cambia lo stato del monopattino (disponibile, non disponibile, in manutenzione) vengono aggiornate anche le liste di monopattini “disponibili” degli altri utenti (es. User, Employee). va bene

**State**: stato del monopattino.

**Strategy**: Pagamento (carta credito, debito, buoni)

**Template Method**: persone -> user, emp, op

**Adapter**: …

**Bridge**: (ereditarietà)

**Composite**: …

**Decorator**: …filtro?

**Facade**: interfaccia unificata.

**Flyweight**: …

**Proxy**: Comunicazione tramite socket

User, Emp e Op comunicano con un server attraverso delle socket.  
Le informazioni permanenti saranno salvati su un DB.

massimo 10 pagine dove si spiega il progetto.

Factory Pattern: creazione di una nuova persona, in base a che si passa si crea un nuovo User, Employee o Operator. (?)  
Forse si, al momento di registrazione.  
Gli utenti possono creare account autonomamente, mentre quelli degli emp e degli op devono essere creati da un altro employee.

<https://www.youtube.com/watch?v=J1QU_R4MQQc>

CoR: noleggio monopattino -> Handler se il monopattino è disponibile, Handler se non è disponibile.  
IDEA2: Handler dipende dal tipo di utente

(vedere Scooter, Hub, ecc…)

Strategy: ricarica del CREDITO tramite carta di credito, carta di debito, buoni.  
(vedi es. Robot. Il robot sarebbe il credito. move sarebbe addCredit).

Memento: ripristina ultima operazione sul credito.  
SI ha 1 ora di tempo, a meno che non sia stato già usato il credito.

IDEA2: Annulla noleggio monopattino (entro 90 secondi)

Observer: quando si cambia lo stato del monopattino (disponibile, non disponibile) vengono aggiornate anche le liste di monopattini “disponibili” degli altri utenti (es. User, Employee).  
La mappa è il soggetto

Mediator: …

Singleton: DB?

Proxy: Mappa -> è un oggetto costoso, quindi ne vogliamo una versione semplificata.  
DB: https://www.youtube.com/watch?v=AB0gaAg9jwI

FORSE MEGLIO NON IMPLEMENTARE la mappa. Solo coordinate.

ProtectionProxy per DB.  
https://refactoring.guru/design-patterns/proxy

Sia Noleggio che abbonamento hanno un COSTO, quindi stessa interfaccia.

Altro Handler per SearchScooters

Payment Handler

**1. Factory Pattern: V**

**2. Cor:**

**3. Strategy:**

**4. Memento**

**5. Proxy**

**6. Mediator**

Lo scooter si deve anche posare.

Interfacce grafiche

Login

Registration

…

**DATABASE**

Installare mysql installer community -> da qui installare Server e Workbench.

video riferimento tutorial: https://www.youtube.com/watch?v=fUK94jOFwBc

Creato un server, creeremo il db attraverso una query.  
**create database** nome;

Possiamo usare il database con:  
**use** sql\_workbench  
da questo punto, possiamo creare tabelle, ecc.

vedere il progetto: jdbcdemo; per reference.

**Esempio Connessione database con Java**:  
<https://www.youtube.com/watch?v=2i4t-SL1VsU>

rimuovere addPerson e removePerson da Employee.  
Non necessario.

aggiungere getPerson (getUser/getEmp/getOp)

Non c’è bisogno di aggiungere robe (tipo carte) nello stesso momento.  
Li aggiungiamo quando facciamo tipo getCards.

NELLA PRESENTAZIONE:

aggiungi video dimostrativo con GUI.

Operazioni di inserimento/rimozione/prelevamento tuple:

1. inserimento persona; V
2. rimozione persona; V
3. recupero dati da persona; V (vedi anche getInfo);
4. aggiornare credito;
5. ottieni carte; V
6. aggiungi carta; V
7. rimuovi carta; V
8. imposta abbonamento;
9. aggiungi Rental;
10. ottieni info utente;
11. ottieni feedback (in base al monopattino);
12. aggiungi scooter;
13. rimuovi scooter;
14. disattiva scooter;
15. ottieni info operatore;
16. assegna scooter a operatore;
17. rimuove scooter da operatore;
18. ottieni scooter da gestire;
19. …