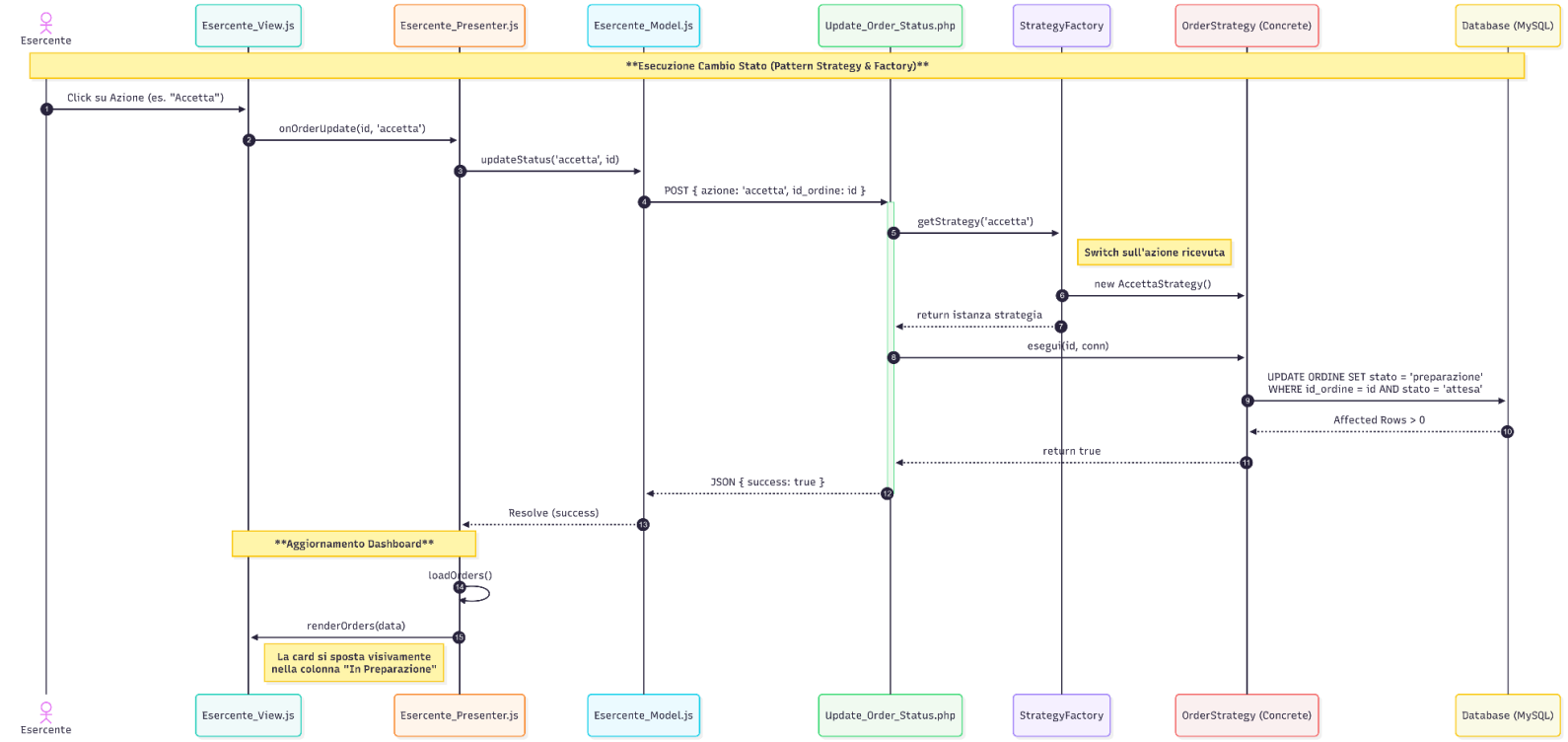
Diagramma di Sequenza - Implementazione Architetturale (Pattern Strategy & Factory) (Task 3.5)

Il diagramma illustra come il sistema gestisce una richiesta di cambio stato (es. da "In Attesa" a "In Preparazione") separando la logica decisionale dalla logica di esecuzione SQL.

1. **Azione Utente**: L'esercente interagisce con la dashboard (es. clicca su "Accetta"). La `View` cattura l'evento e lo delega al `Presenter`.
2. **Richiesta di Aggiornamento**: Il `Presenter` coordina l'operazione invocando il `Model` per effettuare la chiamata asincrona verso il server.
3. **Inoltro al Controller**: Il `Model` invia una richiesta `POST` all'API `Update_Order_Status.php`, specificando l'azione richiesta e l'ID dell'ordine.
4. **Risoluzione della Strategia**: Il controller interroga la `StrategyFactory` passandogli la stringa dell'azione. La Factory istanzia dinamicamente la classe concreta appropriata (es. `AccettaStrategy`, `ProntoStrategy`, ecc.) e la restituisce al controller.
5. **Esecuzione della Business Logic**: Il controller invoca il metodo `esegui()` sulla strategia ottenuta. Ogni strategia incapsula la propria query SQL specifica e le proprie regole di validazione (come il controllo dello stato attuale sul DB).
6. **Persistenza dei Dati**: La strategia esegue l'aggiornamento sul Database. Se l'operazione ha successo (affected rows > 0), il risultato positivo risale la catena fino al frontend.
7. **Sincronizzazione UI**: Al termine dell'operazione, il `Presenter` forza un nuovo ciclo di caricamento dati (`loadOrders`) per riflettere immediatamente lo spostamento dell'ordine nella colonna corretta della Kanban.


Esercente
Esercente_View.js
Esercente_Presenter.js
Esercente_Model.js
Update_Order_Status.php
StrategyFactory
OrderStrategy (Concrete)
Database (MySQL)
Esecuzione Cambio Stato (Pattern Strategy & Factory)
1
Click su Azione (es. "Accetta")
2
onOrderUpdate(id, 'accetta')
3
updateStatus('accetta', id)
4
POST { azione: 'accetta', id_ordine: id }
5
getStrategy('accetta')
Switch sull'azione ricevuta
6
new AccettaStrategy()
7
return istanza strategia
8
esegui(id, conn)
9
UPDATE ORDINE SET stato = 'preparazione'
WHERE id_ordine = id AND stato = 'attesa'
10
Affected Rows > 0
11
return true
12
JSON { success: true }
13
Resolve (success)
Aggiornamento Dashboard
14
loadOrders()
15
renderOrders(data)
La card si sposta visivamente
nella colonna "In Preparazione"
Esercente
Esercente_View.js
Esercente_Presenter.js
Esercente_Model.js
Update_Order_Status.php
StrategyFactory
OrderStrategy (Concrete)
Database (MySQL)