Academia de Studii Economice din București

Facultatea de Cibernetică, Statistică și Informatică Economică



**Modele de proiectare a aplicațiilor de întreprindere**

**-Platformă organizare evenimente-**

Profesor coordonator, Studenti,

Prof. Univ. Dr. Pocatilu Paul Munteanu Cristian

Palade Gabriel

Rotaru Mihaela

București

2024

Cuprins

[Introducere 3](#_Toc155813385)

[Design Patterns 3](#_Toc155813386)

[State 3](#_Toc155813387)

[Observer 5](#_Toc155813388)

[Abstract Factory 6](#_Toc155813389)

[Strategy 8](#_Toc155813390)

# Introducere

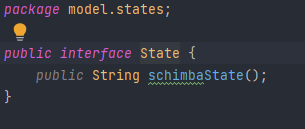
În era digitală contemporană, organizarea de evenimente devine tot mai frecventă și necesită soluții eficiente. Acest proiect își propune să dezvolte o aplicație robustă bazată pe Java, destinată facilitării procesului de gestionare a evenimentelor între organizatori și participanți. Aplicația va oferi o platformă ușor de utilizat pentru programare și comunicare între echipele de organizatori și participanții la evenimente. Scopul principal este să creeze un instrument eficient și intuitiv, menit să facă procesul de gestionare a evenimentelor mai accesibil și simplificat atât pentru organizatori, cât și pentru participanți.

Totodata in realizarea aplicatiei, am folosit urmatoarele design patterns: AbstractFactory, Observer, State, iar aplicatia a fost structurata in pachete, fiecare pachet fiind structurat sub forma arhitecturi MVC.

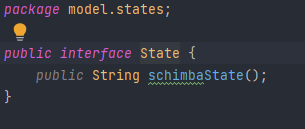
# Design Patterns

## State

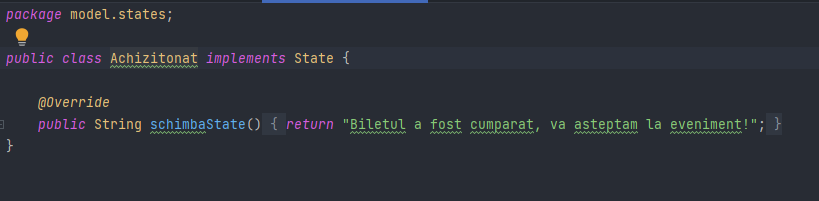
Acest design pattern a fost implementat în pachetul "state".

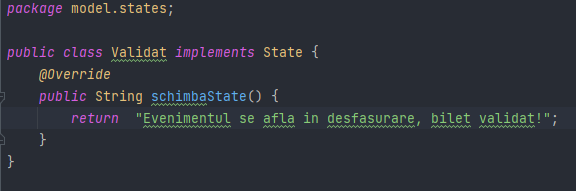


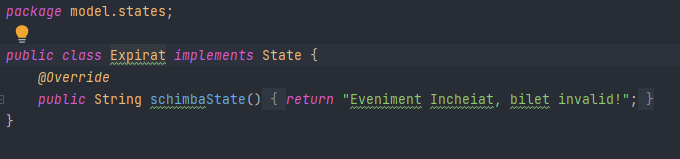
În design pattern, acesta este folosit pentru a schimba mai multe stări ale în care se află un bilet. Astfel, biletul poate avea mai multe stări, cum ar fi: ACHIZIȚIONAT, VALIDAT și EXPIRAT. Pentru a face acest lucru posibil, am utilizat o interfață numită State, care are rolul de a schimba stările biletului cu ajutorul metodei String schimbareState().



De asemenea, au fost create și trei clase denumite Achiziționat, Validat și Expirat care implementează interfata de mai sus. În fiecare clasă este implementată metoda din interfata, iar în funcție de caz, s-a adăugat și un text corespunzător.

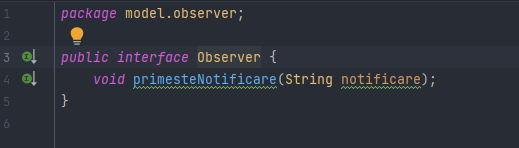






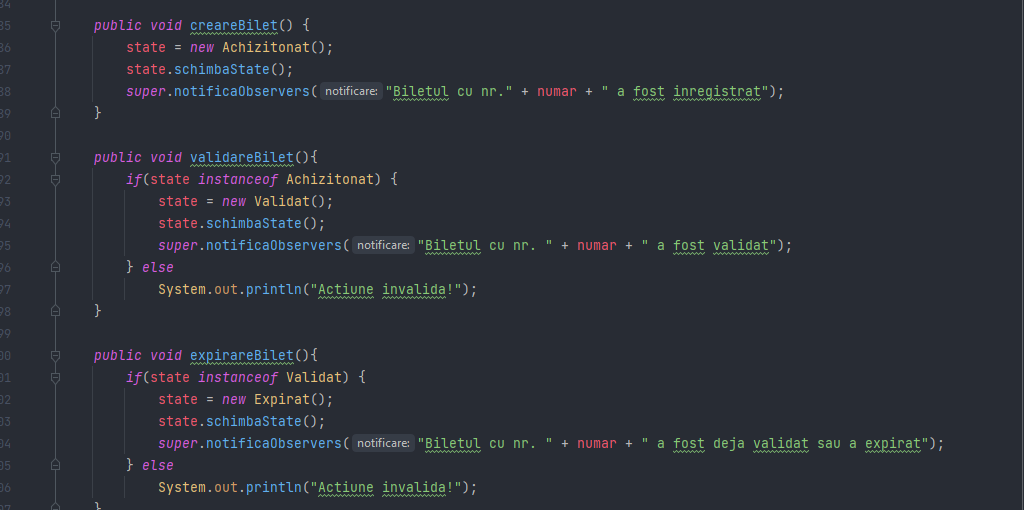
## Observer

Cu acest design pattern, am reușit să notificăm atunci când starile biletului au fost schimbate. În implementarea lui, s-a utilizat o clasă abstractă numită ManagerNotificări și o interfață Observer. În interfață se află o metodă care are denumirea primesteNotificari(String notificare) prin care, mai departe, se poate modifica notificarea.



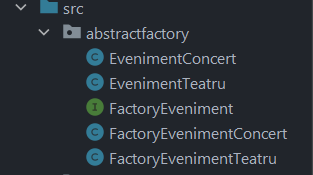


În imaginea de mai jos se poate vedea o utilizare a acestui design pattern în cod.

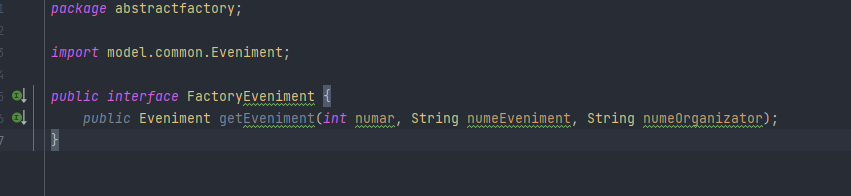


## Abstract Factory

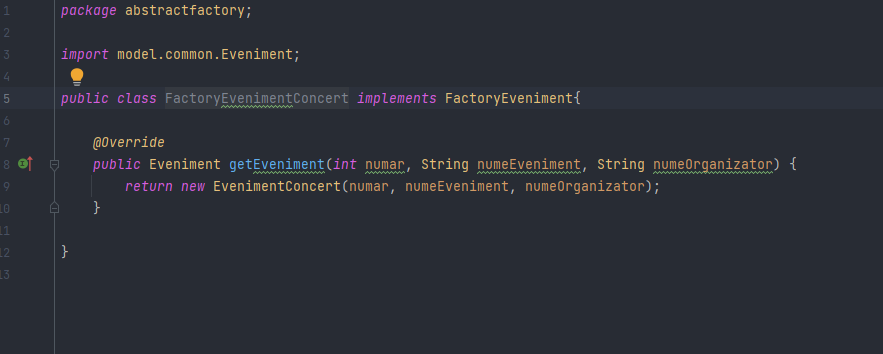
Modelul Abstract Factory constă dintr-o clasă abstractă “factory” care definește o interfață pentru crearea obiectelor și un set de clase “factory” concrete care implementează interfața pentru a crea obiecte specifice. Clasele care folosesc obiectele create de fabrică nu trebuie să știe care clasă “factory” concretă creează obiectele, trebuie doar să cunoască interfața clasei “factory”.

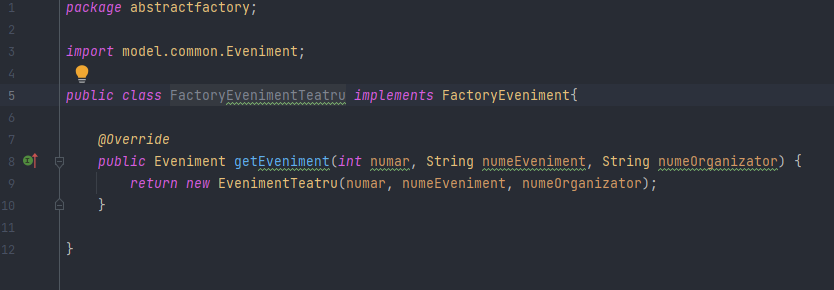


În cazul nostru, clasa abstractă este Eveniment, iar interfata utilizată este FactoryEveniment.

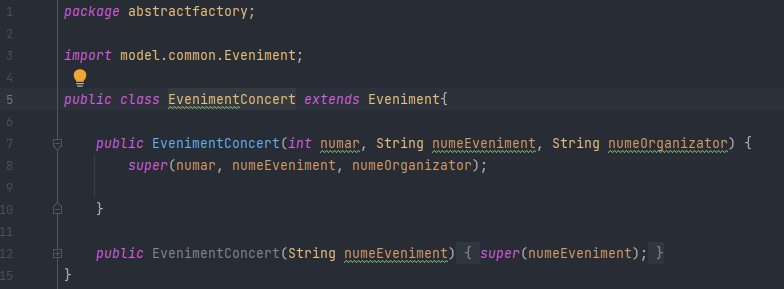


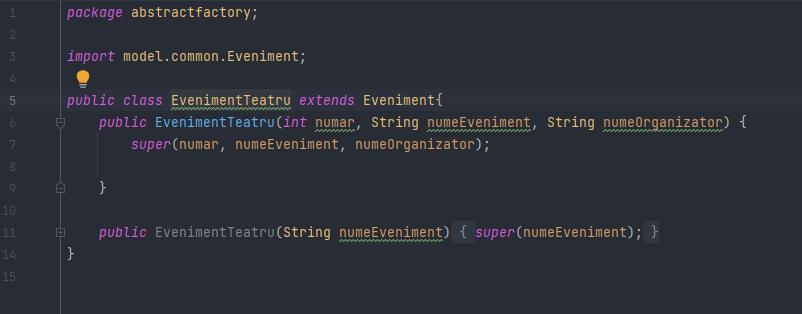
Clasele care implementează interfata FactoryEveniment sunt FactoryEvenimentConcert și FactoryEvenimentTeatru, unde se creează obiectele de tip teatru și concert.





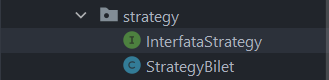
Clasele EvenimentConcert și EvenimentTeatru extind clasa principală de evenimente, care este Eveniment.





## Strategy

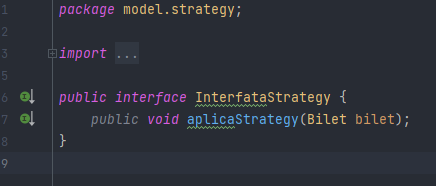
Acest design pattern a fost implementat în pachetul strategy.



Interfata folosita in acest design pattern este InterfataStrategy care definește interfața comună pentru toate strategiile, în acest caz interfața are o singură metodă public void aplicaStrategy(Bilet bilet) care este folosită pentru a aplica strategia schimbarii pretului unui bilet.

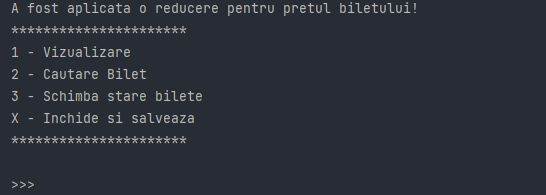
Clasa StrategyBilet este o clasă de strategie concretă care implementează interfața InterfataStrategy, această clasă este responsabilă de aplicarea strategiei de reducerea pretului unui bilet care este mai mare de 400 de lei.





# Rularea aplicatiei

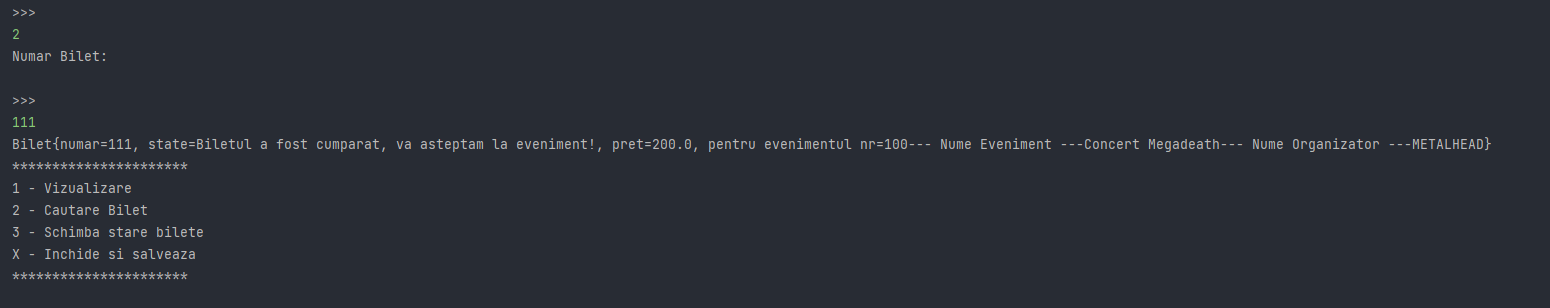
În momentul în care rulăm aplicatia apare în consolă un meniu după care putem să alegem ce putem să facem mai departe în aplicație.



Pentru acțiunea 1 se afiseaza biletele si evenimentele aferente ficarui bilet dupî cum se poate observa și în imaginea de mai jos.



Pentru acțiunea 2 se cere introducerea numarului unui bilet apoi se afisează respectivul bilet.



Pentru actiunea 3 cu se poate schimba starile biletului, iar cu actiunea X se salveaza datele care au fost schimbate.

