PATERNI PONAŠANJA

Observer Pattern

Observer patern omogućava da objekti budu obaviješteni o promjenama stanja drugog objekta. Ovaj patern je koristan kada postoji potreba za obavještavanjem više objekata o promjenama koje su se dogodile u subjektu.

Primjer: U našem sistemu možemo koristiti Observer patern za obavještavanje korisnika o promjenama statusa njihove rezervacije karata.

Strategy Pattern

Strategy patern omogućava definiranje porodica algoritama, enkapsuliranje svakog od njih i njihovu zamjenu prema potrebi. Ovo je korisno kada postoji potreba za mijenjanjem algoritama u toku izvršavanja programa.

Primjer: U našem sistemu možemo koristiti Strategy patern za različite načine obračuna popusta na karte.

Command Pattern

Command patern omogućava enkapsulaciju zahtjeva kao objekta, omogućavajući parametrijizaciju klijenata sa različitim zahtjevima, stavljanje zahtjeva u redove ili zapisivanje operacija.

Primjena: Može se koristiti za upravljanje različitim akcijama u korisničkom interfejsu (npr. poništavanje i ponavljanje akcija rezervacije karata).

Chain of Responsibility Pattern

Chain of Responsibility patern omogućava da se zahtjev prosljeđuje kroz lanac obrađivača dok ga jedan od njih ne obradi.

Primjena: Može se koristiti za obradu različitih zahtjeva korisnika, kao što su zahtjevi za povrat novca ili reklamacije.

Mediator Pattern

Mediator patern omogućava da objekti komuniciraju preko posrednika, smanjujući time međusobne zavisnosti između njih.

Primjena: Može se koristiti za koordinaciju komunikacije između različitih modula sistema, kao što su moduli za rezervaciju karata, plaćanje, i korisničku podršku.

State Pattern

State patern omogućava objektima da mijenjaju svoje ponašanje kada im se promijeni stanje.

Primjena: Može se koristiti za upravljanje različitim stanjima rezervacije karata (npr. kreirana, potvrđena, otkazana).

Template Method Pattern

Template Method patern definira skeleton algoritma u metodi, ostavljajući implementaciju koraka podklasama.

Primjena: Može se koristiti za definisanje opšteg procesa obrade narudžbi, gdje specifični koraci kao što su validacija plaćanja ili slanje potvrde mogu biti definirani u podklasama.

Visitor Pattern

Visitor patern omogućava definiranje nove operacije bez mijenjanja klasa objekata nad kojima se operacija izvršava.

Primjena: Može se koristiti za izvođenje različitih operacija nad kolekcijom objekata, kao što su generisanje izvještaja o prodaji karata.

Zaključak

Dodavanje paterna ponašanja u postojeći sistem za online rezervaciju i prodaju karata donosi mnoge prednosti, uključujući povećanu fleksibilnost, lakšu održivost i proširivost sistema. Primjena Observer i Strategy paterna pomaže u rješavanju specifičnih problema vezanih za obavještavanje korisnika i fleksibilnost algoritama popusta. Ostali paterni ponašanja mogu se primijeniti u različitim dijelovima sistema kako bi se dodatno poboljšala njegova funkcionalnost i efikasnost. Na taj način, sistem postaje modularniji, lakši za proširenje i održavanje.