SOLID Principi

Single Responsibility Principle
(Princip pojedinačne odgovornosti)

Ovaj princip kaže da svaka klasa treba imati samo jedan razlog za promjenu, tj. da bude odgovorna za jednu funkcionalnost.

- Klasa ***Korisnik*** zadužena je isključivo za podatke korisnika (ime, prezime, email, šifra) i nije zadužena za narudžbe, plaćanja niti validaciju proizvoda.
- Klasa ***Plaćanje*** se fokusira isključivo na podatke transakcije (status, datum, transakcijski ID), dok se sve o proizvodima nalazi u drugim klasama.
- Klasa ***ValidacijaProizvoda*** se bavi samo statusom i stanjem proizvoda, a ne prikazivanjem, narudžbom ili plaćanjem.

Dakle, klase imaju jasno odvojene odgovornosti i nema preklapanja funkcionalnosti, čime se SRP u potpunosti poštuje.

Open Closed Principle
(Otvoreno- zatvoreni princip)

Klase trebaju biti otvorene za proširenje, ali zatvorene za modifikaciju.

U našem sistemu ovaj princip je zadovoljen kroz:

- Korištenje enumeracija kao što su ***Kategorija*** i ***Status***, koje se mogu proširivati bez mijenjanja postojeće logike klasa.
- Atribut *parametri* u klasama kao što su ***Proizvod*** i ***Personalizacija,*** omogućavaju dodavanje dodatnih karakteristika bez izmjene strukture.

OCP je ispunjen jer se funkcionalnosti mogu proširivati novim klasama i vrijednostima bez direktne izmjene postojećih klasa.

Liskov Substitution Principle
(Liskov princip zamjene)

Izvedene klase treba da mogu zamijeniti svoje bazne klase bez narušavanja funkcionalnosti.

U našem dijagramu **nije primijenjeno nasljeđivanje** između klasa. Sve klase su nezavisne i ne nasljeđuju zajedničke atribute ili metode iz baznih klasa.

Interface Segregation Principle
(Princip izoliranja interfejsa)

Klase ne bi trebale biti primorane da implementiraju metode koje neće koristiti.

U našem sistemu **nije korišten nijedan interfejs**. Sve klase imaju svoju funkcionalnost direktno implementiranu, bez apstraktnih slojeva ili interfejsa koji bi ih ograničavali.

Dependency Inversion Principle

a) *Moduli visokog nivoa ne bi trebali ovisiti od modula niskog nivoa, oba bi trebalo da ovise od apstrakcija.*
b) *Moduli ne bi trebali ovisiti od detalja. Detalji bi trebali biti ovisni od apsrakcija.*

U dijagramu nisu korišteni interfejsi ni apstraktni slojevi, pa se Dependency Inversion Principle formalno ne primjenjuje, ali samim tim ni ne dolazi do njegovog kršenja, te se može smatrati ispunjenim.