KARDINALNOSTI:

- Svaki Korisnik koristi tačno jedan Račun
- Svaka skupina prodavača (Prodavači) sadrži jednog ili više Prodavača
- Svaki Prodavač ima tačno jednu Statistiku prodaje
- Svaki Prodavač u svojoj ponudi ima 0 ili više proizvoda
- Svaki Proizvod ima tačno jednu Statistiku
- Svaka Trgovina sadrži jedan ili više Proizvoda
- Svaka Narudžba sadrži jedan ili više proizvoda
- Svaka Narudžba ima tačno jednog naručioca (Kupca)
- Svaki Kupac u svojoj historijiKupovine sadrži 0 ili više Proizvoda
- Svaki Kupac u listi Narudžbi sadrži 0 ili više Narudžbi
- Svaka skupina kupaca (Kupci) sadrži jednog ili više Kupaca

SOLID PRINCIPI:

- ✓ Klase Korisnik, Prodavač, Kupac, Korisnici

Što se tiče klase Korisnik, ovdje je ispoštovan SRP (Single Responsibility Principe: Svaka klasa treba imati samo jednu ulogu). Ovaj princip, dakle, zahtijeva da svaka klasa ima samo jednu odgovornost, odnosno da klasa vrši samo jedan tip akcija kako ne bi ovisila o prevelikom broju konkretnih implementacija, što se odnosi na to da je ova klasa zadužena samo za funkcionalnosti vezane za profil korisnika, dakle ispunjava zahtjeve SRP da klasa treba imati samo jednu ulogu.

Također, klasa je otvorena za nadogradnje, ali zatvorena za modifikacije, čime je ispunjen OCP (Open/Closed Principle: Klasa treba biti otvorena za nadogradnje, ali zatvorena za modifikacije) : Moguće je da potrebe korisnika budu zahtijevale nove nadogradnje (da se dodaju još neke metode ili atributi prema potrebama nadogradnje klase Korisnik), ali ove već postojeće metode i atributi klase, nakon što se ispravno implementiraju, neće trebati mijenjati, jer obuhvataju osnovne funkcionalnosti.

Ovdje se može primjetiti i da je ispoštovan LSP (Liskov Substitution Principle: Svaka osnovna klasa treba biti zamjenjiva svim svojim podtipovima bez da to utječe na ispravnost rada programa), zbog toga što je bazna klasa definisana kao apstraktna, tako da će se, u suštini, uvijek „zamjenjivati“ klasama Kupac i Prodavač.

Također, može se primjetiti da je ispoštovan i DIP (Dependency Inversion Principle: Sistem klasa i njegovo funkcionisanje treba ovisiti o apstrakcijama, a ne o konkretnim implementacijama), tj. Ovaj princip zahtijeva da pri nasljeđivanju od strane više klasa bazna klasa uvijek bude apstraktna. Razlog za ovo je što je teško koordinisati veliki broj naslijeđenih klasa i konkretnu baznu klasu ukoliko ista nije apstraktna, a da pritom kod bude čitak i jednostavan za razumijevanje. Mi smo to upravo postigli definisanjem bazne klase Korisnik apstraktnom, nakon što smo odlučili da će iz nje biti naslijeđeno više od jedne klase (Kupac i Prodavač).

Klasa Kupac (i kasnije klasa Gost) će implementirati interfejs PregledProizvoda(*) sa metodama: PogledajOpisProizvoda, PogledajCijenu, PogledajDetalje ... (veza Realizacija: se odnosi na korištenje operacija interfejsa od strane neke klase).

Ovdje je ispoštovan ISP (Interface Segregation Principle: Bolje je imati više specifičnih interfejsa, nego jedan generalizovani). Ovaj princip zahtijeva da i svi interfejsi zadovoljavaju princip S, odnosno da svaki interfejs obavlja samo jednu vrstu akcija, a ovaj interfejs zadužen je samo za akcije vezane za pregled proizvoda, tj. akcije vezane za jednu klasu (klasu Proizvod).

Kod kontenerske klase Korisnici, korišten je apstraktni tip, što znači da je dodatno ispunjen ISP (zavisi od apstrakcija, a ne o konkretnim implementacijama).

Dakle, možemo zaključiti da je ovdje ispunjeno svih 5 SOLID principa.

- ✓ Klasa Račun

1. Single Responsibility Principe: Klasa račun zaista ima samo jednu odgovornost; Princip S zahtijeva da svaka klasa ima samo jednu odgovornost, odnosno da klasa vrši samo jedan tip akcija kako ne bi ovisila o prevelikom broju konkretnih implementacija: ova klasa vrši samo jedan tip akcija vezanih za bankovni račun korisnika i njene metode se uglavnom tiču get/set atributa vezanih za račun korisnika ili eventualno slanje tih podataka banki.

2. Open/Closed Principle: Klasa treba biti otvorena za nadogradnje, ali zatvorena za modifikacije; Klasa račun je otvorena za nadogradnju, ukoliko se pojave potrebe za još nekim metodama ili atributima koji se tiču bankovnog računa korisnika, ali osnovni atributi i metode pokrivaju osnovne funkcionalnosti, stoga ovu klasu ne bi trebalo modifikovati u budućnosti.

3. Liskov Substitution Principle: Svaka osnovna klasa treba biti zamjenjiva svim svojim podtipovima bez da to utječe na ispravnost rada programa;

Ovdje nije realizovano nadljeđivanje, pa nema smisla ispitivati ispunjavanje LSP.

4. Interface Segregation Principle: Bolje je imati više specifičnih interfejsa, nego jedan generalizovani – ovdje nisu realizovani interfejsi

5. Dependency Inversion Principle: Sistem klasa i njegovo funkcionisanje treba ovisiti o apstrakcijama, a ne o konkretnim implementacijama.

- ✓ Klase Proizvod i Trgovina

Ovdje je ispunjen SRP SOLID princip, jer klasa Proizvod ima samo jednu odgovornost, vezanu za eventualne promjene proizvoda, te get/set metode vezane za Proizvod.

Također je ispunjen OCP Princip, jer su moguće nadogradnje klase, ali su trenutno navedene metode i atributi dovoljni da bi se pokrile osnovne funkcionalnosti, koje neće trebati modificirati.

Realizovan je interfejs PregledProizvoda (jer će ga različite nepovezane klase implementirati: klasa Kupac i klasa Gost) : PogledajOpisProizvoda, PogledajCijenu, PogledajDetalje... , pa je ovdje je ispunjen ISP Princip.

Ovaj interfejs je osmišljen zato što se nudi mogućnost da se artikli pregledaju i ako je neko pristupio kao gost, a ne samo kao već prijavljeni kupac. Tada takav klijent ima mogućnost da pretražuje artikle i vidi njihove cijene, opise i detalje, ali nema opciju naručivanja (kupovine).

Ostale principe nema smisla ispitivati, jer ovdje nije realizovano nasljeđivanje (LSP i DIP).

- ✓ Klasa Narudžba

<u>Ovdje su ispunjeni sljedeći SOLID principi:</u>

SRP (ova klasa ima samo jednu odgovornost vezanu za realizaciju narudžbe proizvoda),
OCP (otvorena je za eventualne nadogradnje, a zatvorena za modifikacije ukoliko se pojave potrebe za tim);

Ostale principe nema smisla ispitivati (OCP, LSP i DIP), jer ovdje nije realizovano nasljeđivanje, niti su realizovani neki interfejsi.

- ✓ Klasa Statistika

<u>Ovdje su ispunjeni sljedeći SOLID principi:</u>
SRP (Klasa Statistika ima samo jednu odgovornost – zadužena je isključivo za statističke proračune prodaje nego prodajnog profila),

OCP (otvorena je za eventualne nadogradnje, ukoliko se ukaže potreba za nekim novim funkcionalnostima vezanim za statistiku, npr. Da se doda metoda koja preračunava dnevnu statistiku prodaje, koja istražuje kada je najveća potražnja za određenim proizvodima, i sl...), ali su osnovne metode i atributi dovoljni da bi pokrili osnovne funkcionalnosti, koje neće trebati modifikovati.

Ostale principe nema smisla ispitivati (OCP, LSP i DIP), jer ovdje nije realizovano nasljeđivanje, niti su realizovani neki interfejsi.