S – Single Responsibility Principle (SRP)

Svaka klasa treba da ima jednu i samo jednu odgovornost.

Primjena u modelu:

- Klasa **Korisnik** upravlja isključivo korisničkim podacima (identitet, pristup, status).
- Klasa **Pitanje** opisuje isključivo entitet pitanja u QnA sistemu.
- Klasa **Odgovor** sadrži samo podatke vezane za odgovore.
- Klasa **Poruka** obrađuje samo privatnu komunikaciju između korisnika.
- Klasa **Notifikacija** modeluje samo obavještenja korisnicima.

Zaključak: Nema miješanja odgovornosti – svaka klasa ima jasno definisanu i jedinstvenu svrhu.

O – Open/Closed Principle (OCP)

Klase treba da budu otvorene za proširivanje, a zatvorene za modifikaciju.

Primjena u modelu:

- Korišćenje **enumeracija** (npr. `Uloga`, `Status`, `Tip dostignuća`) omogućava lako proširivanje bez izmjene strukture klase.
- Sistemi kao što su **Dostignuće** i **Notifikacija** mogu se proširiti novim tipovima bez potrebe za izmjenom postojećih klasa.

Zaključak: Sistem je dizajniran tako da omogućava dodavanje novih funkcionalnosti (npr. novih uloga, statusa, tipova dostignuća) bez izmjena postojećeg koda.

L – Liskov Substitution Principle (LSP)

Objekti izvedenih klasa treba da mogu zamijeniti objekte baznih klasa bez narušavanja sistema.

Primjena u modelu:

- Iako hijerarhija nasleđivanja nije eksplicitna u ovom modelu, dizajn omogućava buduće proširenje kroz apstrakcije (npr. `Komentar` može biti nadklasa za `KomentarNaPitanje`, `KomentarNaOdgovor`).
- Relacije kao što su `VezanoZa: object` kod komentara impliciraju potencijal za polimorfizam (interfejs za entitete koji mogu imati komentare).

Zaključak: Model podržava potencijalno nasleđivanje bez narušavanja logike, poštujući LSP.

I – Interface Segregation Principle (ISP)

Klijenti ne treba da budu primorani da zavise od interfejsa koje ne koriste.

Primjena u modelu:

- Svaka klasa sadrži samo atribute relevantne za njen kontekst (npr. klasa **OcjenaPredmeta** ne nasleđuje metode/atribute koje koristi klasa **Poruka**).
- Buduće implementacije interfejsa (npr. `IKomentabilan`, `IRejtingEntitet`) mogu se ograničiti samo na relevantne klase.

Zaključak: Nema prisilnog korišćenja funkcionalnosti koje nisu potrebne, što zadovoljava ISP.

D – Dependency Inversion Principle (DIP)

Moduli višeg nivoa ne treba da zavise od modula nižeg nivoa; oboje treba da zavise od apstrakcija.

Primjena u modelu:

- Iako je model statički i bez metoda, relacije kao što su `Komentar -> VezanoZa: object` impliciraju zavisnost od apstrakcije, a ne od konkretne klase (`Pitanje` ili `Odgovor`).
- Sistemski servisi (npr. notifikacije, rejting) mogu se razviti kao interfejsi čije implementacije ne zavise direktno od modela.

Zaključak: Model je dizajniran tako da se logika može osloniti na apstraktne tipove ili interfejse, što omogućava visoku fleksibilnost.