Zadaća 1

**Izvještaj o Inspekciji Koda klase „namjesten stan“**

**Inspekcija strukture programskog rješenja**

*Ova lista stavlja fokus na potencijalne probleme u strukturi programskog koda, analizi koda na visokom nivou i poštovanju standarda.*

Kod je napisan u skladu sa važećim standardima kodiranja.

Stil kodiranja je konzistentan u cijelom programskom rješenju.

Kod je ispravno formatiran.

U kodu nema funkcija koje se ne pozivaju ni na jednom mjestu.

Nema nedostižnih linija koda.

Nema bespotrebnog implementiranja funkcija koje mogu biti zamijenjene postojećim bibliotekama.

U kodu nema mnogo implementiranih funkcija, ali ovo bi se moglo razmotriti na osnovu potreba i daljnjeg razvoja projekta.

U kodu nema ponavljanja koje može biti zamijenjeno jedinstvenom funkcijom.

Memorija se koristi na efikasan način.

Nema korištenja *magičnih brojeva* i konstanti bez korištenja varijabli.

Nema previše dugih i kompleksnih blokova koda.

**Inspekcija dizajna programskog rješenja**

*Ova lista stavlja fokus na potencijalne probleme u poštovanju objektno-orijentisanih principa, SOLID principa i dizajn pattern-a u okviru programskog rješenja.*

Svaka klasa ima malu kompleksnost i jedan tip operacija i zaduženja.

Klasa **NamjestenStan** ima relativno mali broj metoda i čini se da obavlja zadatke koji se odnose na definirane atribute stanova.

Klase su prilagodljive budućim promjenama.

Svi objekti izvedenih klasa zamjenjivi su svojim osnovnim klasama.

Interfejsi su jednostavni, s malim brojem funkcija.

Klasa **NamjestenStan** nema eksplicitno definiran interfejs, ali implementira dva metoda koja nisu komplicirana.

Dubina nasljeđivanja nije velika.

Klijent može jednostavno pristupati objektima kontejnerskih klasa, bez potrebe definisanja detalja gradivnih dijelova klase.

Ovisno o upotrebi u aplikaciji, potrebno je provjeriti kako klijenti pristupaju objektima klase **NamjestenStan**.

U slučaju potrebe ponovnog korištenja većeg broja istih objekata, objekti se ne instanciraju više puta.

Instanciranje kontejnerske klase vrši se samo jednom.

Sigurnost aplikacije osigurana je putem *proxy*-a.

Ovisno o zahtjevima aplikacije, sigurnost aplikacije može zahtijevati dodatne slojeve i sigurnosne mehanizme.

**Inspekcija varijabli i izraza programskog rješenja**

*Ova lista stavlja fokus na potencijalne probleme u strukturi koda na visokom nivou, uključujući varijable i izraze u kodu.*

Sve varijable imaju imena koja odgovaraju njihovoj namjeni.

Koristi se jedan stil imenovanja varijabli.

Nema varijabli koje se ne koriste.

Nema neosiguranih potencijalnih dijeljenja s nulom.

Operator **=** ne koristi se u logičkim izrazima.

**Inspekcija petlji i grananja programskog rješenja**

*Ova lista stavlja fokus na potencijalne probleme u petljama i grananjima u kodu.*

Nema praznih niti nedostižnih blokova koda.

U *if* blokovima testiraju se češći scenariji.

Svi *switch* iskazi imaju definisan *default* slučaj.

U klasi **NamjestenStan**, nema switch iskaza, pa se ovaj element ne primjenjuje.

Sve petlje imaju uslov završetka.

Nema velikog broja gniježdenja petlji.

U petljama nema koda koji se može izvršiti izvan petlje.

**Inspekcija memorijskih operacija programskog rješenja**

*Ova lista stavlja fokus na potencijalne probleme u korištenju memorije te konekciji s bazama podataka, vanjskim uređajima i korištenjem file-ova u kodu.*

Sve varijable koje koriste indeksiranje su inicijalizirane prije korištenja.

Sva alocirana memorija dealocira se prije završetka izvršavanja.

Pri radu s vanjskim uređajima, postoji provjera za *timeou0t*.

U klasi **NamjestenStan**, nema izričite provjere za timeout prilikom rada s vanjskim uređajima ili vanjskim resursima. Ovisi o kontekstu aplikacije da li ovo treba dodatno obraditi.

Prije pokušaja modificiranja *file*-ova, provjerava se da li oni postoje.

U klasi **NamjestenStan**, nema provjere postojanja datoteka prije pokušaja njihove modifikacije. To može biti važno ovisno o zahtjevima aplikacije i treba uzeti u obzir ako je primjenjivo.

Nakon završetka transakcije, konekcija s bazom podataka se uvijek zatvara.

U klasi **NamjestenStan**, nema izravne veze s bazama podataka ili otvaranja/zatvaranja konekcija. Ovo je specifično za aplikacije koje interagiraju s bazama podataka, pa se ovaj element ne primjenjuje na ovu klasu.

**Inspekcija dokumentacije programskog rješenja**

*Ova lista stavlja fokus na potencijalne probleme u razumljivosti i jednostavnosti dokumentovanja koda.*

Svi kompleksni dijelovi koda posjeduju komentare.

Dijelovi koda podijeljeni su u regije.

Klasa **NamjestenStan** ne koristi regije za grupiranje dijelova koda.

Metode klasa imaju svoje opise.

U klasi **NamjestenStan**, metode kao što su **Ispisi** i **ObracunajCijenuNajma** imaju implementirane funkcionalnosti, ali nisu opisane komentarima. Preporučljivo je dodati komentare koji objašnjavaju svrhu tih metoda i kako rade.

U cijelom rješenju koristi se jedan stil komentarisanja koda.

U programskom rješenju nisu korišćeni komentari, ali ako bi se koristili, preporučljivo je održavati dosljedan stil komentarisanja koda kroz sve klase i dijelove koda.

***Informacije o timu koji vrši inspekciju koda***

*Popuniti informacije o članovima tima koji vrši inspekciju.*

Ime i prezime, broj indexa: Eldar Muratović, 145-ST

Zaduženje: Inspekcija Program i Stan klase

Predmet inspekcije: Checklista

Ime i prezime, broj indexa: Vedad Ribo, 154-ST

Zaduženje: Inspekcija Nenamjesten Stan klase

Predmet inspekcije: Checklista

Ime i prezime, broj indexa: Hamza Bunar, 165-ST

Zaduženje: Inspekcija Namjesten Stan

Predmet inspekcije: Checklista

***Izvještaj o metrikama grešaka***

Ukupan broj pronađenih grešaka: Click here to enter text.

Normirani broj grešaka: Click here to enter text.

Broj grešaka po LOC: Click here to enter text.

Broj normiranih grešaka po LOC: Click here to enter text.

Efikasnost otkrivanja grešaka: Click here to enter text.

Normirana efikasnost otkrivanja grešaka: Click here to enter text.