Zadaća 1

**Izvještaj o Inspekciji Koda klase „Program“**

**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 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.

Klase su prilagodljive budućim promjenama.

Svi objekti izvedenih klasa zamjenjivi su svojim osnovnim klasama.

Interfejsi su jednostavni, s malim brojem funkcija.

Dubina nasljeđivanja nije velika.

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

• Postoji potencijalna mogućnost da klijenti moraju znati više o unutarnjim detaljima klasa kako bi ih koristili. Možda bi se moglo koristiti više principa enkapsulacije kako bi bilo pristupačnije.

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.

• Sigurnost aplikacije nije provjerena u primjeru koda.

**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.

• U kodu bi moglo biti potencijalno nepotrebnih varijabli, ali to ovisi o budućim proširenjima aplikacije. Preporučuje se pregled i uklanjanje nepotrebnih varijabli ako se utvrdi da nisu potrebne.

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 kodu se ne koristi switch iskaz, pa se ovaj kriterij 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.

• U kodu nema eksplicitne dealokacije memorije jer C# koristi sustav za upravljanje memorijom (Garbage Collector) za automatsku dealokaciju memorije. Ovo je uobičajeni pristup u C#.

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

• U kodu nema provjere za timeout prilikom rada s vanjskim uređajima. Ovisno o situaciji, dodavanje mehanizma za rukovanje timeout-om može biti korisno kako bi se izbjegli problemi s blokiranjem.

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

• U kodu nema provjere postojanja file-ova prije njihove modifikacije. Ovo je pristup koji bi mogao poboljšati sigurnost i stabilnost aplikacije, posebno u situacijama kada je rad s file-ovima bitan.

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

**Inspekcija dokumentacije programskog rješenja**

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

Svi kompleksni dijelovi koda posjeduju komentare.

• Nema komentara.

Dijelovi koda podijeljeni su u regije.

Metode klasa imaju svoje opise.

U cijelom rješenju koristi se jedan stil komentarisanja 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