Laboratorijska vježba br. 1

Izvještaj o Inspekciji Koda

Check liste inspekcije koda

Označiti stavke na listama za inspekciju koje su ispunjene, nakon vršenja inspekcije koda. Za stavke koje se ne označe potrebno je navesti detaljne informacije o greškama u nastavku.

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.
☐ 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.

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.
☒ 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*.
☐ Prije pokušaja modificiranja *file*-ova, provjerava se da li oni postoje.
☐ 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.
☐ 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: Muhammed Šehić, 227-ST
Zaduženje: Sistem posudbe i vraćanja knjiga
Predmet inspekcije: Kood Modula 'BorrowRepository.cs', 'BorrowRecord.cs' i 'BorrowService.cs'

Ime i prezime, broj indexa: Click here to enter text.
Zaduženje: Click here to enter text.
Predmet inspekcije: Click here to enter text.

Izvještaj o pronađenim greškama

Popuniti informacije o pronađenim greškama, te kategorijama u koje spadaju. Lokacija greške u modulu podrazumijeva file i linije koda u kojima se greška nalazi.

Br.	*Check Lista*	*Tip*	*Opis*	*Lokacija*	*Ozbiljnost*
1.	Dizajn	Magic Numbers	Vrijednosti `BorrowDaysLimit` i `FinePerDay` bile su hardcodirane prije dodavanja konstanti; preporučeno ih je centralizovati u static config.	BorrowService.cs	Niska
2.	Greške / Exception Handling	Nedostatak specifičnih izuzetaka	Korišteni su generički `Exception`, umjesto custom tipova (`UserNotFoundException`, `BookUnavailableException`).	BorrowService.cs	Srednja
3.	Struktura	Nedostatak validacije inputa	Metode `Posudi()` i `Vrati()` ne provjeravaju negativne ID vrijednosti ili nevalidan format prije čitanja iz repozitorija.	BorrowService.cs	Srednja
4.	Dokumentacija	Nema XML komentara	Javne metode servisa i repozitorija nisu dokumentovane, pa je teže razumjeti tok i svrhu parametara.	BorrowService.cs / BorrowRepository.cs	Niska

5.	Logovanje	Nedostatak audit logova	Servis ne bilježi posudbu, vraćanje, kašnjenja — u realnom sistemu ovo je obavezno.	BorrowService.cs	Srednja
6.	Performanse	Korištenje `Find()` na listi knjiga	`GetAllBooks().Find(...)` prolazi cijelu listu; za veliku biblioteku trebalo bi imati dictionary ili indeks.	LibraryInventory.cs	Srednja
7.	Logika	Kašnjenje se računa samo u danima	`(record.ReturnedDate - record.DueDate).Days` zanemaruje sate, pa korisnik može kasniti 23h a dobiti kaznu 0 KM.	BorrowService.cs	Niska
8.					
9.					
10.					

Izvještaj o metrikama grešaka

Ukupan broj pronađenih grešaka: 7

Normirani broj grešaka: 14

Broj grešaka po LOC: 0.014

Broj normiranih grešaka po LOC: 0.014

Efikasnost otkrivanja grešaka: 90%

Normirana efikasnost otkrivanja grešaka: 1.26%