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.
Moj komentar: Struktura projekta modularna (Data, Models, Services), poštuje standarde i SOLID principe. Moguće poboljšanje: odvojiti validaciju emaila u posebnu pomoćnu metodu radi lakšeg testiranja.

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.
Moj komentar: Proxy ili autentifikacijski sloj nije implementiran, što bi bilo poželjno u kasnijim verzijama (posebno za administratore).

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.
Moj komentar: Petlje korištene samo u `PrikaziSve()` metodi — jednostavne i sigurne, bez rizika od beskonačnih iteracija.

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.
Moj komentar: Nema direktnog rada s fajlovima ni bazom — sve se odvija u memoriji. Ako se kasnije doda trajno čuvanje podataka, treba dodati kontrole pristupa fajlu/bazi.

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.

Moj komentar: Regije nisu korištene, ali kod je dovoljno jasan. Preporuka: dodati XML komentare za svaku javnu metodu u `UserService` i `UserRepository`.

Informacije o timu koji vrši inspekciju koda

Popuniti informacije o članovima tima koji vrši inspekciju.

Ime i prezime, broj indexa: Kemal Mešić, 236-ST
Zaduženje: Implementacija modula 1. Upravljanje korisnicma
Predmet inspekcije: Kod modula „UserService“ , „UserRepository“ i „User.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	Nedostatak sigurnosnog sloja	Nema provjere autentifikacija prije izmjene/brisanja korisnika	UserService.cs	Srednja
2.	Struktura	Validacija podataka	Validacija emaila i lozinke mogla bi biti u posebnoj metodi	UserService.cs	Srednja
3.	Dokumentacija	Nedostatak XML komentara	Nema dokumentacija za javne metode	UserRepository.cs / UserService.cs	Niska
4.	Dizajn	Ponavljanje koda	Kod za ažuriranje korisnika ponavlja se ručno za svako polje	UserRepository.cs (Update metoda)	Niska
5.	Memorijske operacije	Nedostaje kontrola pristupa fajlovima	Nema provjere postojanja fajla (ako se uvede file sistem)	N/A	Niska
6.					
7.					
8.					
9.					
10.					

Izvještaj o metrikama grešaka

Ukupan broj pronađenih grešaka: 5

Normirani broj grešaka: 5/3 člana = 1.67 po članu

Broj grešaka po LOC: 5 / 230 = 0.021

Broj normiranih grešaka po LOC: 0.007

Efikasnost otkrivanja grešaka: 100% u okviru modula

Normirana efikasnost otkrivanja grešaka: visoka (većina manjih nepravilnosti)

Zaključak o inspekciji

Tokom inspekcije utvrđeno je da je kod:

- modularan, čitljiv i lako proširiv,
- implementiran uz poštovanje osnovnih OOP principa,
- bez ozbiljnih funkcionalnih ili logičkih grešaka.

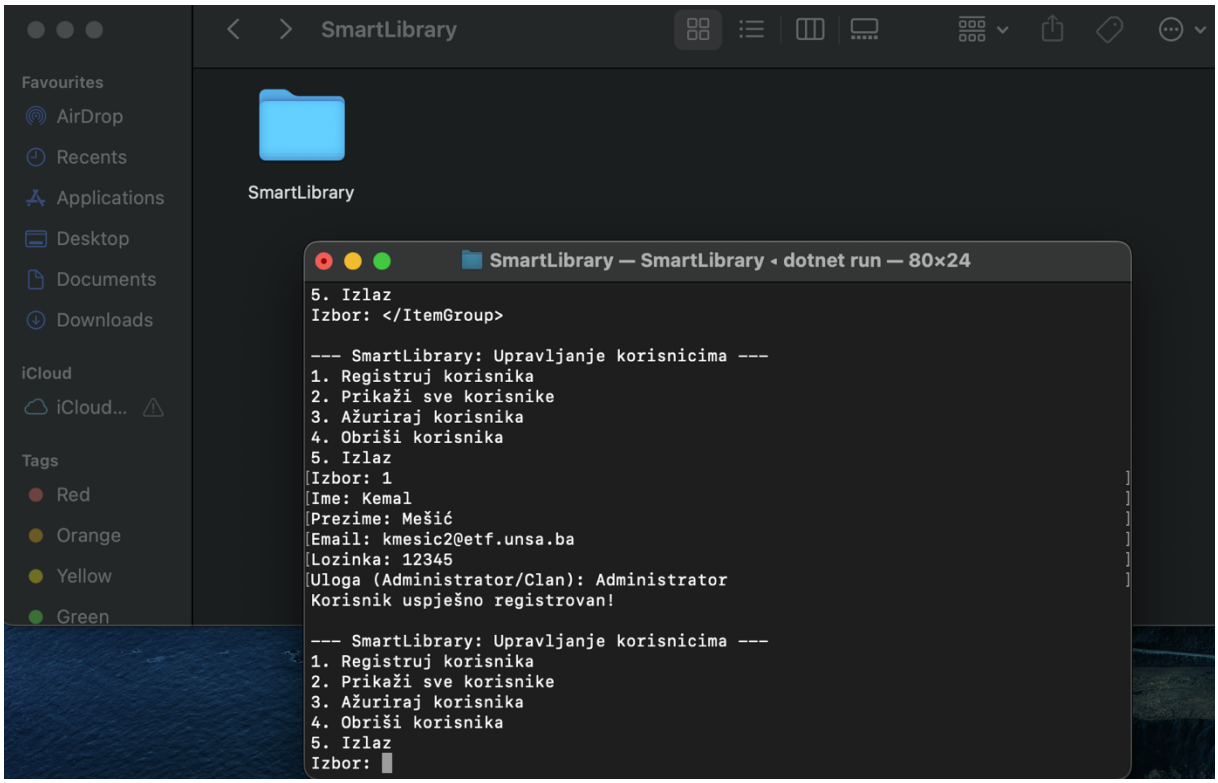
Preporučuje se:

1. Dodavanje XML komentara radi generisanja dokumentacije.
2. Implementacija osnovne autentifikacije i zaštite administrativnih funkcija.
3. Refaktor validacije i ažuriranja korisnika radi bolje testabilnosti.

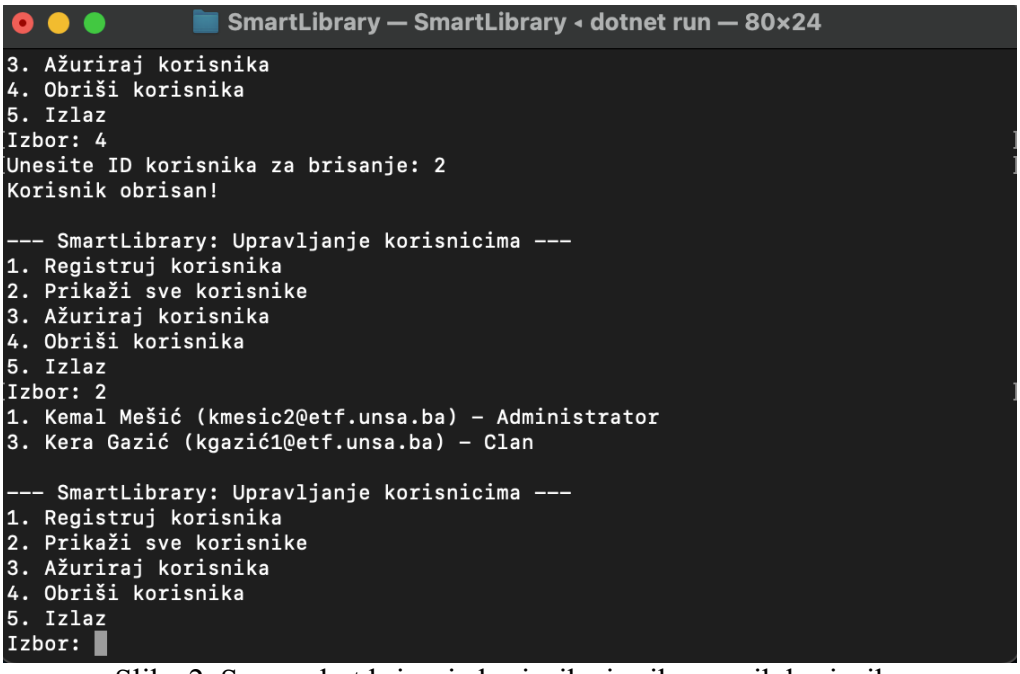
📸 Dokaz o inspekciji (screenshoti i proces)

Uz izvještaj prilažem sljedeće:

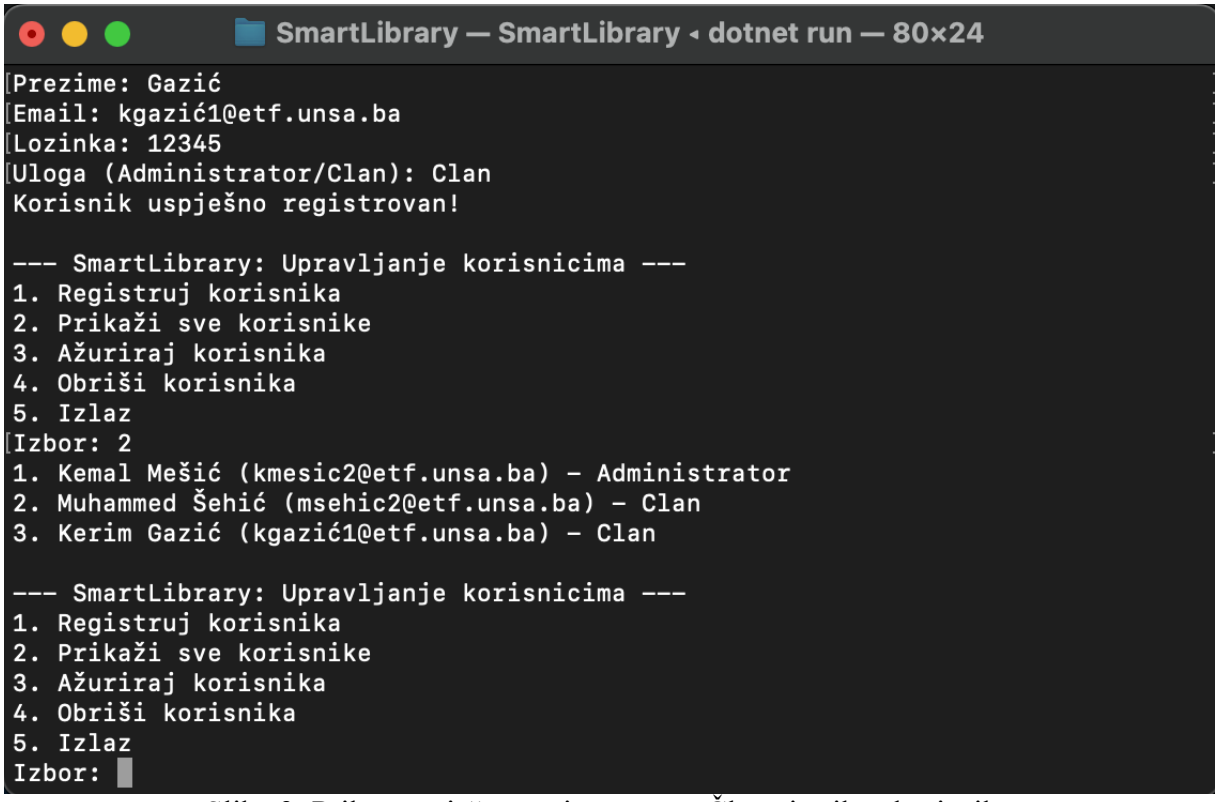
- Screenshote pokretanja aplikacije
- pregleda koda u VS Code-u (označene linije sa komentarima).



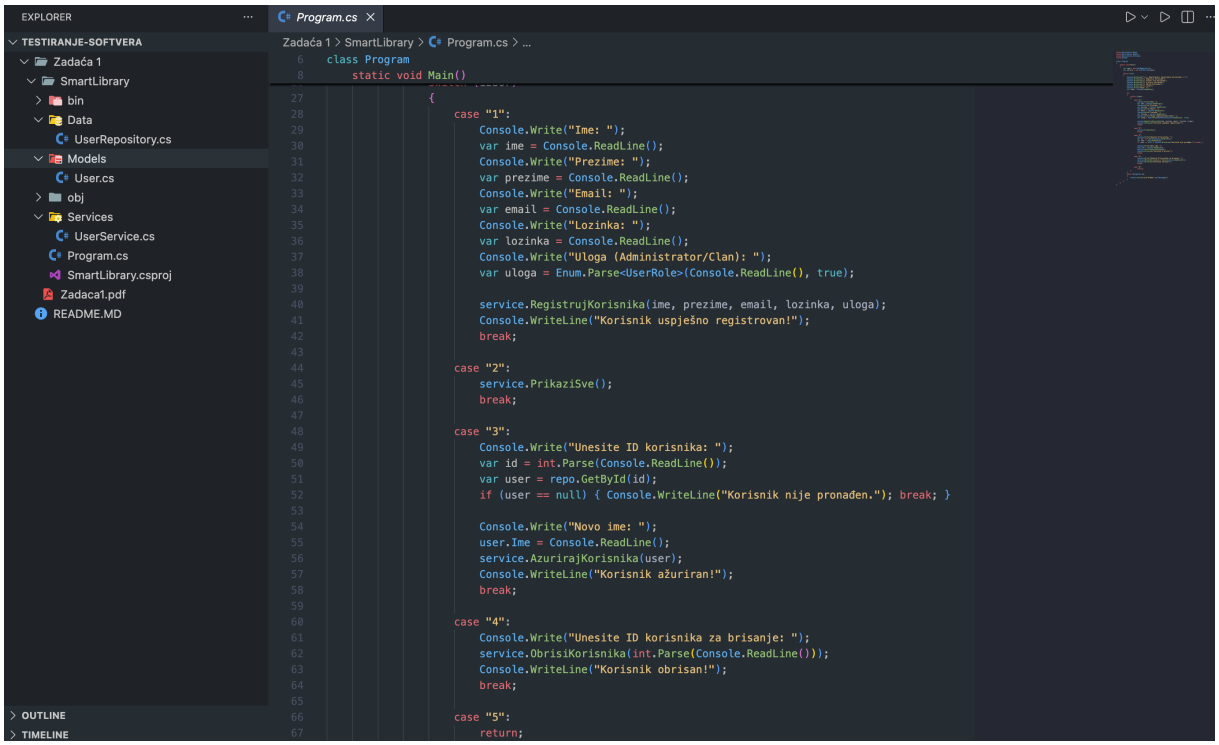
Slika 1: Screenshot registraacije korisnika



Slika 2. Screenshot brisanja korisnika i prikaza svih korisnika



Slika 3: Prikaz uspješno registrovanog Člana i prikaz korisnika



Slika 4: Prikaz koda iza ispis podataka na osnovu odabira korisnika aplikacije