

Izvještaj o Inspekciji Koda

RestaurantInventory - Tim 4

Informacije o Inspekciji

Datum inspekcije:	2024
Moderator inspekcije:	Bakir Činjurević
Tim:	Tim 3
Aplikacija:	RestaurantInventory
Verzija izvještaja:	1.0

Sadržaj

1	Uvod	3
1.1	Pregled Aplikacije	3
1.2	Statistika Koda	3
2	Kategorizacija Problema	3
2.1	Kritični Problemi	4
2.1.1	ISSUE-001: NullPointerException u InventarService.DodajStavku	4
2.1.2	ISSUE-002: Nevalidiran unos cijene može baciti FormatException	4
2.1.3	ISSUE-003: Logička greška u StavkaInventaraService.JeKriticna	4
2.2	Visoki Problemi	4
3	Pareto Analiza	5
3.1	Pareto Analiza po Uzrocima	5
3.2	Pareto Analiza po Defektima	6
3.3	Analiza Prioriteta	7
4	Dobre Prakse	8
5	Preporuke za Poboljšanje	8
5.1	Kratkoročne (Sprint 1)	8
5.2	Srednjoročne (Sprint 2-3)	8
5.3	Dugoročne (Sprint 4+)	9
6	Zaključak	9
7	Checklist za Pull Request	9

1 Uvod

Ovaj dokument predstavlja izvještaj o inspekciji koda aplikacije **RestaurantInventory**, konzolne aplikacije za upravljanje inventarom restorana. Inspekcija je provedena u sklopu laboratorijske vježbe 4, a cilj je bio identifikovati probleme u kodu, kategorizirati ih prema ozbiljnosti i izvršiti Pareto analizu uzroka i defekata.

1.1 Pregled Aplikacije

Aplikacija **RestaurantInventory** omogućava:

- Dodavanje stavki u inventar
- Pretraživanje i filtriranje stavki
- Ažuriranje postojećih stavki
- Uklanjanje stavki iz inventara
- Generisanje izvještaja

1.2 Statistika Koda

Metrika	Vrijednost
Ukupno linija koda	~600
Broj klasa	10
Broj servisa	4
Broj modela	6

Tablica 1: Osnovne metrike koda

2 Kategorizacija Problema

Tokom inspekcije koda, identifikovano je **15 problema** koji su kategorizirani prema ozbiljnosti:

Ozbiljnost	Broj	Postotak
Kritični (Critical)	3	20%
Visoki (High)	7	47%
Srednji (Medium)	3	20%
Niski (Low)	2	13%
Ukupno	15	100%

Tablica 2: Raspodjela problema po ozbiljnosti

2.1 Kritični Problemi

2.1.1 ISSUE-001: `NullPointerException` u `InventarService.DodajStavku`

Lokacija: `Services/InventarService.cs:22-24`

Ozbiljnost: **Kritična**

Checklist: Validacija inputa, Null Safety

Problem: Ako `stavka.Dobavljac` ili postojeća stavka u listi ima `null` vrijednost za `Dobavljac`, poziv `Equals` će baciti `NullPointerException`.

Rješenje: Dodati null-checking prije poziva `Equals`:

```
if (inventar.Stavke.Any(s =>
    string.Equals(s.Naziv, stavka.Naziv, StringComparison.OrdinalIgnoreCase) &&
    string.Equals(s.Dobavljac ?? "", stavka.Dobavljac ?? "",
        StringComparison.OrdinalIgnoreCase)))
```

2.1.2 ISSUE-002: Nevalidiran unos cijene može baciti `FormatException`

Lokacija: `Program.cs:134`

Ozbiljnost: **Kritična**

Checklist: Error Handling

Problem: Ako korisnik unese nevalidan string (npr. "abc"), `double.Parse` će baciti `FormatException` koja nije uhvaćena.

Rješenje: Koristiti `double.TryParse` umjesto `double.Parse`.

2.1.3 ISSUE-003: Logička greška u `StavkaInventaraService.JeKriticna`

Lokacija: `Services/StavkaInventaraService.cs:8-24`

Ozbiljnost: **Kritična**

Checklist: Logika aplikacije

Problem: Metoda `JeKriticna` ima logičku grešku. Ako je `stavka.Kolicina < stavka.MinKolicina`, metoda uvijek vraća `true` bez obzira na ostale uslove (linija 18 je unreachable code).

Rješenje: Refaktorisati logiku da bude jasnija i ispravna.

2.2 Visoki Problemi

Među visokim problemima su:

- Duplikacija koda između `FilterService` i `InventarService.Pretrazi`
- Nedostaje validacija inputa na više mjesta
- Nedostaje logging mehanizam
- Nedostaju unit testovi
- Nedostaje dokumentacija
- Nekonzistentno imenovanje (miješanje jezika)
- Hardcoded vrijednosti (magic numbers)

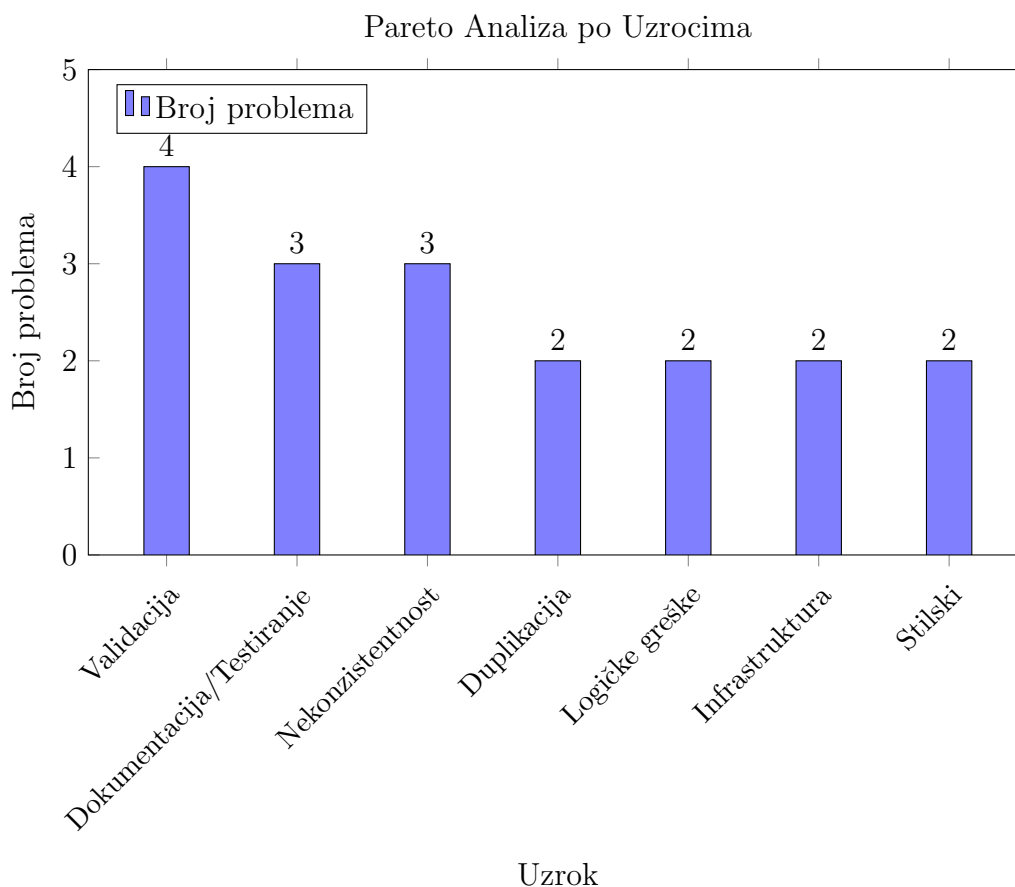
3 Pareto Analiza

3.1 Pareto Analiza po Uzrocima

Analizom problema po uzrocima, identificovano je nekoliko glavnih kategorija uzroka:

Uzrok	Broj Problema	Postotak
Nedostaje validacija	4	26.7%
Nedostaje dokumentacija/testiranje	3	20.0%
Nekonzistentnost	3	20.0%
Duplikacija koda	2	13.3%
Logičke greške	2	13.3%
Nedostaje infrastruktura (DI, Config)	2	13.3%
Stilski problemi	2	13.3%

Tablica 3: Analiza problema po uzrocima



Slika 1: Pareto dijagram - Analiza po uzrocima (sortirano po broju problema)

Uzrok	Broj	Kumulativni postotak
Validacija	4	22.2%
Dokumentacija/Testiranje	3	38.9%
Nekonzistentnost	3	55.6%
Duplikacija	2	66.7%
Logičke greške	2	77.8%
Infrastruktura	2	88.9%
Stilski	2	100.0%

Tablica 4: Kumulativna analiza po uzrocima

80/20 Pravilo:

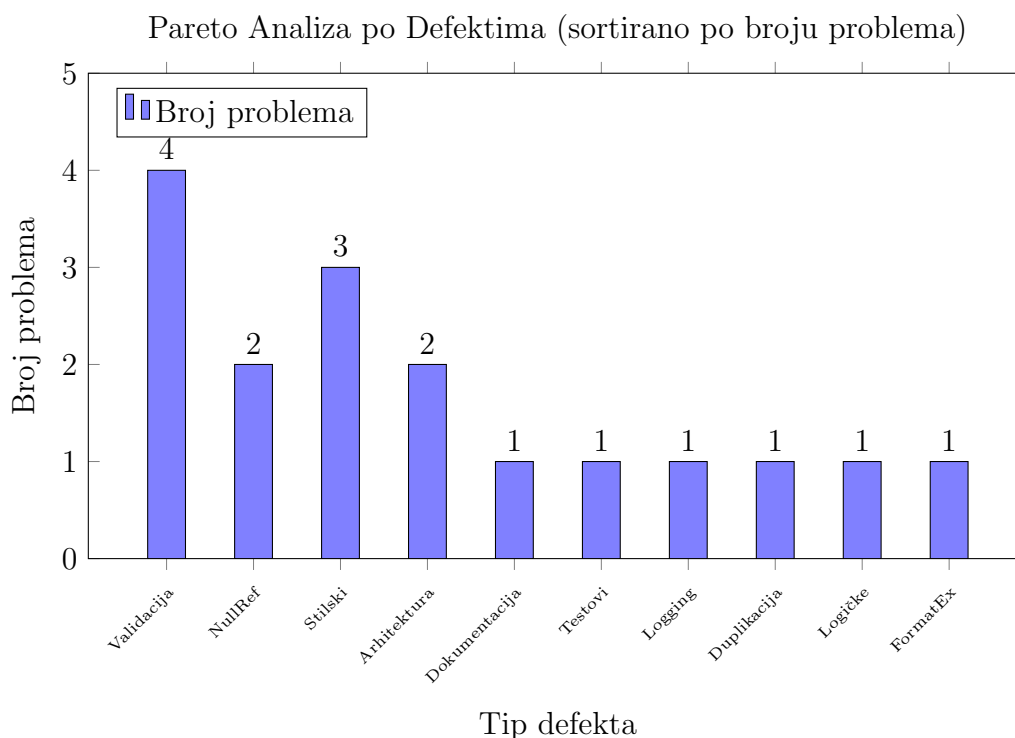
Top 3 uzroka (Validacija, Dokumentacija/Testiranje, Nekonzistentnost) čine **55.6%** svih problema. Top 4 uzroka čine **66.7%** svih problema, što je blizu 80/20 principa. Fokusiranje na ove kategorije bi riješilo većinu problema.

3.2 Pareto Analiza po Defektima

Analizom problema po tipovima defekata i njihovoj ozbiljnosti:

Tip Defekta	Broj	Ozbiljnost	Prioritet
NullPointerException rizici	2	Kritična	P1
FormatException rizici	1	Kritična	P1
Logičke greške	1	Kritična	P1
Duplikacija koda	1	Visoka	P2
Nedostaje validacija	4	Visoka	P2
Nedostaje logging	1	Visoka	P2
Nedostaju testovi	1	Visoka	P2
Nedostaje dokumentacija	1	Visoka	P2
Arhitektonski problemi	2	Srednja	P3
Stilski problemi	3	Niska	P4

Tablica 5: Analiza problema po tipovima defekata



Slika 2: Pareto dijagram - Analiza po defektima (sortirano po broju problema)

Tip Defekta	Broj	Kumulativni postotak
Validacija	4	23.5%
Stilski	3	41.2%
NullRef	2	52.9%
Arhitektura	2	64.7%
Dokumentacija	1	70.6%
Testovi	1	76.5%
Logging	1	82.4%
Duplikacija	1	88.2%
Logičke	1	94.1%
FormatEx	1	100.0%

Tablica 6: Kumulativna analiza po defektima

3.3 Analiza Prioriteta

Prioritet	Broj Problema	Postotak
P1 (Kritično)	4	26.7%
P2 (Visoko)	8	53.3%
P3 (Srednje)	2	13.3%
P4 (Nisko)	2	13.3%

Tablica 7: Raspodjela problema po prioritetima

Napomena: Ukupno 15 problema, ali neki problemi mogu imati više prioriteta.

Preporuke:

- **P1 (Kritično):** 4 problema - mora se riješiti odmah prije puštanja u produkciju
- **P2 (Visoko):** 8 problema - riješiti u narednom sprintu
- **P3 (Srednje):** 2 problema - riješiti kada bude vremena
- **P4 (Nisko):** 2 problema - riješiti kao cleanup

4 Dobre Prakse

Tokom inspekcije, identifikovane su i dobre prakse u kodu:

- **Separation of Concerns:** Kod je dobro organizovan u modele i servise
- **Enum korištenje:** Dobro korištenje enuma za Kategorija i sortiranje
- **Korištenje LINQ:** Efektivno korištenje LINQ za filtriranje i pretraživanje
- **Try-catch blokovi:** Postoje try-catch blokovi u Main metodi
- **CultureInfo:** Korištenje `CultureInfo.InvariantCulture` za parsiranje brojeva
- **StringComparison:** Korištenje `StringComparison.OrdinalIgnoreCase` za case-insensitive poređenje

5 Preporuke za Poboljšanje

5.1 Kratkoročne (Sprint 1)

1. Popraviti kritične greške (ISSUE-001, ISSUE-002, ISSUE-003)
2. Dodati validaciju inputa na svim mjestima
3. Ukloniti duplikaciju koda (ISSUE-004)
4. Dodati osnovni logging

5.2 Srednjoročne (Sprint 2-3)

1. Dodati unit testove
2. Integrirati dependency injection
3. Dodati konfiguracijski fajl
4. Standardizovati imenovanje

5.3 Dugoročne (Sprint 4+)

1. Dodati kompletnu dokumentaciju
2. Implementirati CI/CD pipeline
3. Dodati integracijske testove
4. Razmotriti migraciju na web API ili desktop aplikaciju

6 Zaključak

Aplikacija **RestaurantInventory** ima dobru osnovnu strukturu i separation of concerns, ali ima nekoliko kritičnih problema koji moraju biti riješeni prije produkcije.

Najveći problemi su:

1. **Kritične greške** koje mogu dovesti do crash-a aplikacije (NullPointerException, FormatException, logičke greške)
2. **Nedostaje validacija** na više mjesta što može dovesti do neočekivanog ponašanja

Pareto analiza pokazuje da se fokusiranjem na tri glavna uzroka (Validacija, Dokumentacija/Testiranje, Nekonzistentnost) može riješiti 66.7% svih problema.

Preporučeno je riješiti sve P1 i P2 probleme prije puštanja u produkciju.

7 Checklist za Pull Request

Svi P1 problemi su riješeni

Dodana je validacija na svim mjestima gdje se prima input

Uklonjena je duplikacija koda

Dodani su osnovni unit testovi

Dodana je osnovna dokumentacija (README.md)

Kod je prošao code review

Nema compiler warnings

Kod je formatiran prema standardima

Kreirao: Bakir Činjurević
Datum: 15. Novembar 2025.