**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

**INFORMATIKOS FAKULTETAS**

**PROGRAMŲ SISTEMŲ TESTAVIMAS (T120B162)**

**Statinė kodo analizė**

Atliko: IFF – 4/3 gr. studentas

Dovydas Petrutis

**KAUNAS 2017**

Turinys

[Turinys 2](#_Toc493421109)

[1. Įvadas 3](#_Toc493421110)

[2. Tikslas 3](#_Toc493421111)

[3. Kodo paeržiūra 3](#_Toc493421112)

[4. Kodo peržiūros sąrašas 3](#_Toc493421113)

[5. Statinė kodo analizė su įrankiu 4](#_Toc493421114)

[5.1 Aptiktos klaidos 5](#_Toc493421115)

[6. Naujos statinės kodo analizės taisyklės kūrimas 5](#_Toc493421116)

[7. Išvados 6](#_Toc493421117)

# Įvadas

Statinė kodo analizė yra labai naudinga, nes be didelių programuotojo pastangų, naudojantis tik statinės kodo analizės įrankiais, galima sugauti daug klaidų dar programavimo metu. Taip pat, tai padeda išlaikyti bendrą kodo kokybę ir standartus tarp komandos narių, kas dar labiau pagerina kodo stabilumą.

# Tikslas

Atlikti kodo peržiūrą prieš panaudojant statinės analizės įrankį. Po to atilikti statinę kodo analizę su pasirinktu įrankiu.

# Kodo paeržiūra

Kadangi šį projektą sukūriau pats, tai nuo pat pradžių naudojausi JetBrains ReSharper įrankiu Visual Studio programavimo aplinkai. Jis man padėjo iškarto surasti kodo neatitikimus ir juos sutvarkyti. Dar kartą peržiūrėjęs kelias klases neaptikau jokių neatitikimų iš išvardinto sąrašo.

1. Threads synchronization (missing locks, releases)
2. Uninitialized variables, null pointers. No checks for null pointers
3. Unreachable code paths.
4. SQL injection
5. Missing constructors, missing destructors
6. Invalid access to class fields.
7. Data overflow.
8. Unhandled exceptions

# Kodo peržiūros sąrašas

1. Structure
   1. Does the code conform to coding standards?

Taip.

1. Documentation
   1. Are all comments consistent with the code?

Taip.

1. Variables
   1. Are all variables properly defined with meaningful, consistent, and clear names?

Taip.

* 1. Do all assigned variables have proper type consistency or casting?

Taip.

1. Arithmetic Operations
   1. Does the code avoid comparing floating-point numbers for equality?

Nebuvo ką lyginti.

* 1. Does the code systematically prevent rounding errors?

Nebuvo aritmetinių operacijų.

* 1. Are divisors tested for zero or noise?

Tokių nebuvo.

1. Loops and Branches
   1. Does every case statement have a default?

Nėra switch.

* 1. Does the code in the loop avoid manipulating the index variable or using it upon exit from the loop?

Taip.

1. Defensive Programming
   1. Are indexes, pointers, and subscripts tested against array, record, or file bounds?

Ne.

* 1. Is every memory allocation de-allocated?

Taip.

* 1. Are files checked for existence before attempting to access them?

Nenaudojau išorinių failų.

* 1. Are all files and devices are left in the correct state upon program termination?

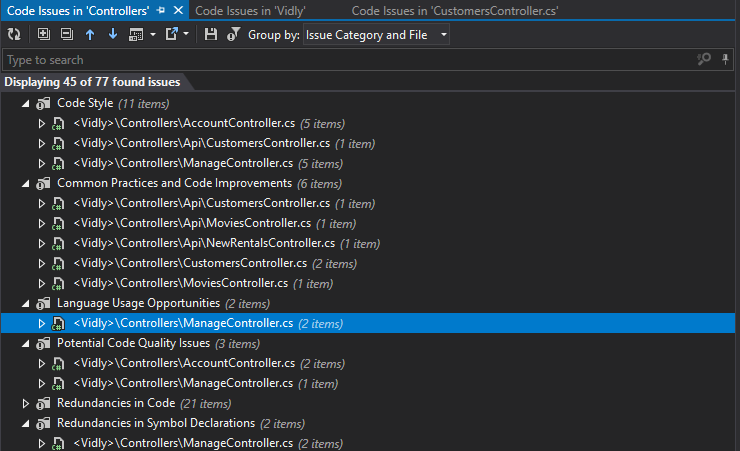
Taip.

# Statinė kodo analizė su įrankiu

Statinei kodo analizei pasinaudojau JetBrains ReSharper įraniu skirtu Visual Studio aplinkai.

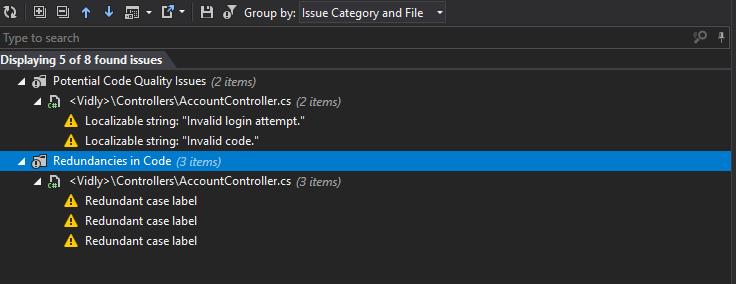
## Aptiktos klaidos

Matome kad įrankis rado nemažai problemų su kodo, kaip kad kodo standartų nesilaikymas, nenaudojimai using sakiniai, neišnaudotos kalbos teikiamos galimybės, kodo pasikartojimai ir pan. (1 pav.)



1 pav. ReSharper statinės kodo analizės rasto klaidos

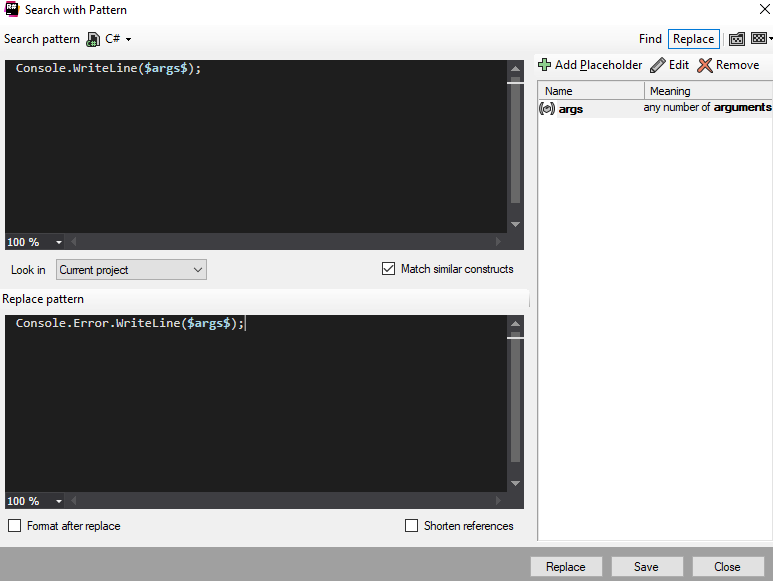
Ištaisius kodo stiliaus klaidas, lieka klaidos metoduose, kurie buvo sugeneruoti automatiškai, tačiau jų taisyti nevalia, nes bandant juos ištaisyti atsiranda dar daugiau neatitikimų ir klaidų.



2 pav. Sutvarkytos klaidos, liko tik klaidos automatiškai sugeneruotuose metoduose

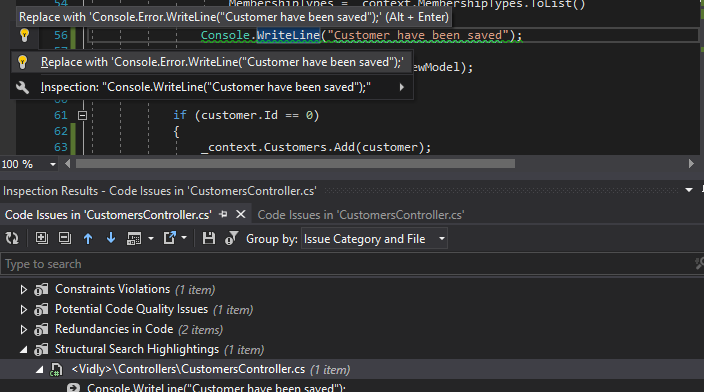
# Naujos statinės kodo analizės taisyklės kūrimas

ReShaper įrankis leidžia nesunkiai apsirašyti kokias tik norite statinės kodo analizės taisykles, kaip pvz. galime tai pamatyti (3 pav.). Taisyklė yra nelogiška, bet ją tokią sukūriau tik dėl demonstracijos.



3 pav. Naujos statinės kodo analizės taisyklės kūrimas

Sukūrus taisyklę, ReSahper visada padeda ją aptikti, taip pat ir paleidus kodo inspektorių ją galime pamatyti ten. (4 pav.) Kaip matome ReSharper įrankis siūlo taisyti šią eilutę, pagal naujai sukurtą taisyklę.



4 pav. Naujos taisyklės aptikimas

# Išvados

Statinės analizės įrankiai programuojant leidžia išlaikyti kodo stilių ir išvengti įvairiausiu klaidų ar neatitikimų, bei leidžia optimizuoti kodą, dar net jo nepaleidus. Mano atveju ReSharper įrankis padėjo išvengti klaidų ir atlikus kodo analizę sutvarkyti kodo stilių ir pagerinti jo kokybę bei skaitomumą. Taip pat jis turi plačias konfigūravimo galimybes, kurios padeda norint sukurti kitas reikalingas taisykles.