

# Engenharia de Software 2 - Relatório TP1

Matheus Flávio Gonçalves Silva - 2020006850

Universidade Federal de Minas Gerais (UFMG)  
Belo Horizonte - MG - Brasil

matheusfgs@ufmg.br

## 1 Membros

Matheus Flávio Gonçalves Silva

## 2 Análise de qualidade com Lizard

### 2.1 Execução da Ferramenta Lizard

```
10 file analyzed.
=====
NLOC    Avg.NLOC  AvgCCN  Avg.token  function_cnt  file
-----
51      24.0     1.0     102.0      1             ./db/Review.py
57      16.0     3.5     164.5      2             ./db/load_classes_from_csv.py
22      7.0      1.0     59.0       1             ./db/User.py
43      7.0      1.5     50.2       4             ./db/Session.py
5       5.0      2.0     28.0       1             ./db/parse_login.py
7       0.0      0.0     0.0        0             ./db/Professor.py
12      0.0      0.0     0.0        0             ./db/Turma.py
8       0.0      0.0     0.0        0             ./db/Disciplina.py
46      0.0      0.0     0.0        0             ./api_responses.py
367     24.2     6.7     211.5     13            ./main.py

=====
No thresholds exceeded (cyclomatic_complexity > 15 or length > 1000 or nloc > 1000000 or parameter_count > 100)
=====
Total nloc  Avg.NLOC  AvgCCN  Avg.token  Fun Cnt  Warning cnt  Fun Rt  nloc Rt
-----
618        18.7    4.7     157.6     22        0         0.00   0.00
```

Figura 1: Saída da execução da ferramenta Lizard no Backend

Conforme visto acima, a maioria das funcionalidades estão localizadas no arquivo **main.py**. Assim sendo, o foco da análise e modificação manter-se-á nas funções presentes nesse arquivo.

## 2.2 Análise da Função mais complexa

Ao analisar o código, a função mais complexa é a função **fetch\_review()**. Esta função está diretamente relacionada com a filtragem e ordenação das reviews de acordo com o semestre, professor, disciplina, classificação, podendo ser dinamicamente alterada de acordo com a preferência do usuário em tempo de uso.

Assim sendo, essa função faz uso de consultas ao banco de dados com base nos filtros fornecidos, além de realizar o processamento dos resultados retornados pelo banco e retorna os dados obtidos como resposta ao frontend para exibição correta das reviews. Trata-se da função mais importante do sistema, afinal, de nada adianta fazer uma review a respeito de qualquer coisa que seja e essas reviews não serem exibidas.

Dentre as estruturas e paradigmas envolvidos na função, temos a manipulação de listas, a utilização de filtros de pesquisa, além da ordenação dinâmica de acordo com a seleção do usuário.

## 3 Adicionais

A partir desse ponto, encontram-se explicações e documentações adicionais que não foram exigidas no PDF segundo o TP. Contudo, por questão de facilidade, foram acrescentadas a partir dessa sessão para maior conforto. Os itens 1 e 3 pedidos na especificação encontram-se a seguir.

## 4 Explicação sobre o sistema

Para esse sistema web é idealizado um fórum de avaliação a respeito de professores da UFMG com dados referentes ao período de 2020/1 a 2023/1, seleção do semestre lecionado, além da validação se o professor deu aula da matéria especificada no semestre selecionado. Além disso, pode-se selecionar se serão feitas avaliações de forma anônima ou sem anonimato de acordo com a escolha do usuário ao criar a postagem.

## 5 Tecnologias

- Quart (Backend)

Foi escolhido o Quart como backend pela facilidade da manipulação da linguagem Python e pela similaridade com o framework Flask utilizado para desenvolvimento Web. De certo modo, pode-se dizer que o Quart é a reimplementação do Flask com a possibilidade de execução de funções assíncronas.

- React (Frontend)

Foi escolhido o React como ferramenta para desenvolvimento do frontend pela maior intimidade com esse framework.

- MongoDB (Banco de Dados)

Para o banco de dados foi escolhido o MongoDB pela possibilidade de utilização do MongoDB Compass facilitar a correção de pequenos problemas, a visualização dos dados e correção do banco. Foram cogitados também os bancos MySQL e SQLite, mas deixados de lado em virtude da experiência de maior facilidade de lidar com o MongoDB.

## 6 Refatoração

Refactoring 1

Refactoring 2