GITHUB

original -> https://github.com/danielsilva83/arquitetura

forked -> https://github.com/FelipeKamimura/arquitetura

Esse questionário deve ser usado como guia durante o primeiro *code review*. Vocês podem fazer outras observações caso achem pertinentes.

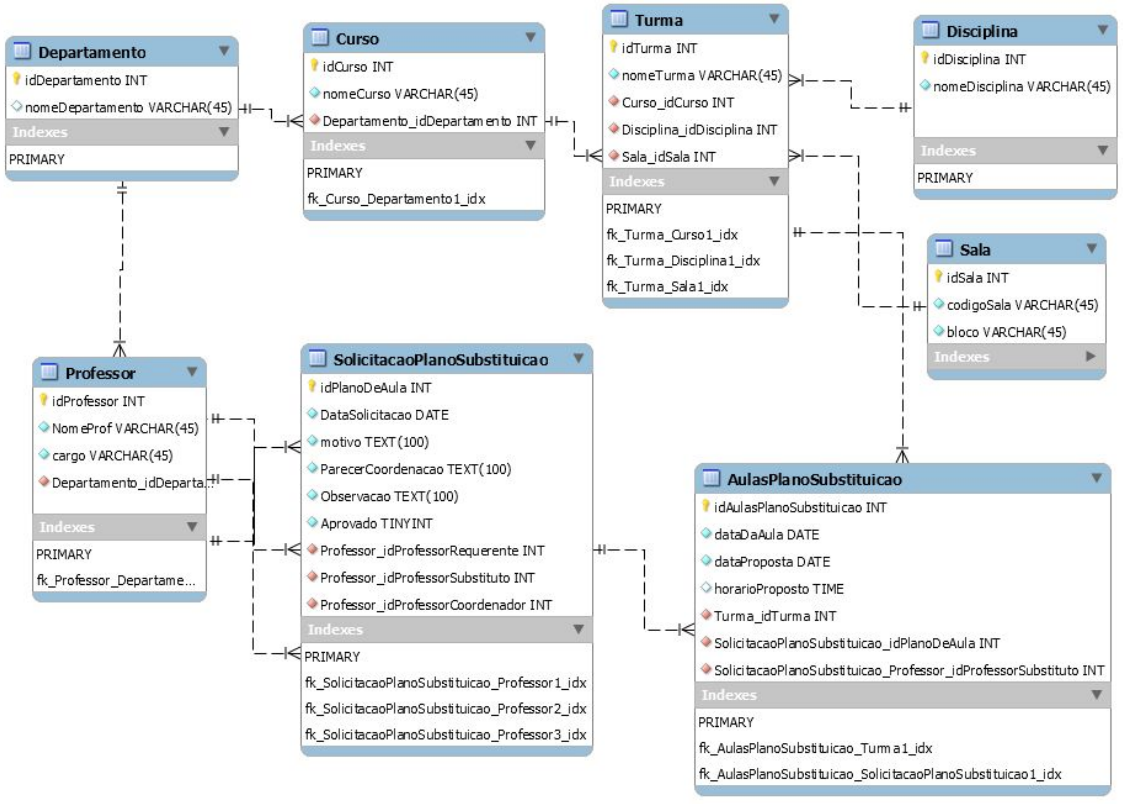
-> Manutenibilidade

- O projeto contém alguma descrição arquitetural de atividades para auxiliar no entendimento das regras de negócio?

Não. O projeto poderia ter um readme ou algum comentário no código explicando a as regras, ou o próprio documento utilizado como base para entender as regras de negócio.

- O projeto contém alguma descrição arquitetural de comunicação/sequência para auxiliar no entendimento do fluxo de comunicação entre as classes?

Sim, possui um modelo de banco de dados que torna possível ver as relações das entidades, que se refletem como classe no desenvolvimento.



- As classes/métodos são curtos o suficiente para facilitar o entendimento?

Sim, são curtos o suficiente, mas os métodos encontrados no controller não são auto explicativos.

- A decomposição da arquitetura em pacotes/classes beneficiar ao máximo a manutenibilidade?

Sim, o programa foi dividido em pacotes (entity, controllers, Project, dao), porém ainda falta algumas estruturas e conexões entre as funções dos pacotes (o software se encontra incompleto), os controllers convergem com a ideia da aplicação em java, utilizando de notações do spring para isso, segundo o desenvolvedor as notações são voltadas para uma aplicação web.

-> Princípios de desenvolvimento

- O projeto usa inversão de dependência de forma adequada?

Não, pois não houve uso de inversão de dependência no projeto, dado que nenhuma abstração foi implementada.

- O projeto usa o princípio *open-closed* quando conveniente?

Não, pois para atender esse princípio é necessária a implementação de classes abstratas, onde as classes poderão sobrescrever os métodos da classe abstrata, facilitando sua manutenção, porém nenhuma abstração foi implementada no projeto.

- O projeto usa o princípio de substituição de Liskov quando conveniente?

Não, na LSP é dito que uma classe filha deve poder substituir uma classe base e no código não há nenhum tipo de herança, apenas do JPA que seria um framework, além que no código onde houve utilização de interfaces não houve o resto da implementação.

- O projeto usa o princípio de segregação de interfaces quanto conveniente?

Sim, há uma segregação das interfaces no pacote DAO do código, onde cada entidade possui uma interface DAO separada para a mesma e não tendo "desperdício" de funções que poderiam ter na interface.

- O projeto evita repetição frequente de código?

Não se aplica. O software possui suas entidades e dao's com as notações do lombok e com a utilização do jpa que evita a digitação com a repetição de métodos para cada um dos objetos. Na pacote dos controllers não

- As rotinas implementadas nos métodos são simples de entender e alterar?

São simples de entender, a dúvida fica quanto ao funcionamento que não está claro (mas cabe também ao conhecimento do grupo que está fazendo o code review, que apesar de entendermos a intenção, desconhecemos a execução do mesmo).

- O acoplamento entre as classes é o menor possível dentro do contexto do projeto?

Isso se aplica às entidades e ao dao.

- As classes estão coesas?

Sim, pois as classes não manipulam dados que não fazem parte de seus respectivos contextos ou que sejam responsabilidade de outras classes.

-> Estruturas arquiteturais

- A estrutura de dados foi implementada corretamente?

A parte do DAO e Entity estão corretos e completos cada possuindo sua relação com o outro.

- A estrutura de camadas foi implementada corretamente?

Não se aplica, pois o controller não liga todo o projeto.

- O Github foi corretamente utilizado para representar/controlar a estrutura de alocação da equipe?

Em partes, em algum momento duas branches divergiram, mas é possível ver dois contribuidores.

-> Padrões de projeto

- Algum padrão de projeto foi usado quando conveniente?

Sim, o DAO para os banco de dados.

- A estrutura do padrão de projeto foi implementada corretamente?

Sim, conforme o padrão do jpa. O que poderia ser melhorado é o esquema das classes, como por exemplo as conexões entre elas.

-> Bibliotecas

- Alguma biblioteca foi usada quando conveniente?

Sim o Lombok, para setar os getters e setters, além dos construtores. E facilitar a utilização dos objetos.

- Algum mecanismo de gerenciamento de bibliotecas foi usado? (ex: Maven)

Foi usado o Maven, que pode ser comprovado pelo arquivo pom.xml

-> Framework

- Algum framework foi usado quando conveniente?

Sim, o jpa para mapear o banco de dados e o springboot para a geração do projeto, é conveniente pois as dependências do programa já são prontas para a

importação, aumenta a produtividade e não livra da necessidade das configurações do spring puro.

- O framework adotado gera algum benefício claramente perceptível?

Sim, facilidade em setar o dao (banco de dados) e importações como o lombok.