**Guia de Implementação**

SUMÁRIO

[1. Introdução 2](#_Toc485801465)

[1.1. Referências 2](#_Toc485801466)

[2. Organização e Estilo do Código 2](#_Toc485801467)

[3. Comentários 2](#_Toc485801468)

[4. Nomeação 2](#_Toc485801469)

[5. Declaração 3](#_Toc485801470)

[6. Reutilização 3](#_Toc485801471)

[7. Tratamento de Erros e de Exceções 3](#_Toc485801472)

[8. teste unitário 3](#_Toc485801473)

[9. Diretrizes Gerais 4](#_Toc485801474)

# Introdução

No back-end foi realizado as etapas de cadastramento, atualização e desativação de usuário que se cadastram em nosso sistema e os projetos publicados pelos usuários. Falta ainda fazer um ambiente remoto de homologação e validações dos dados enviadas pelas as requisições HTTP feitas pelo front-end, também devem ser feitos refinos com filtros nas requisições de dados. As maiores dificuldades estão nas configurações do servidor remoto para hospedagem de uma versão de homologação do nosso sistema.

## Referências

Documentos utilizados como fonte de informações para desenvolvimento do projeto.

* FlyWay Documentation - <https://flywaydb.org/documentation/> - acessado 15/12/2020
* Guide Spring Framework - <https://spring.io/guides> - acessado 15/12/2020
* Swagger - <https://swagger.io/tools/swagger-ui/> - acessado 15/12/2020

# Organização e Estilo do Código

O projeto esta utilizando as convenções Java de codificação, documentação da convenção pode ser acessadas no sítio de www.oracle.com

# Comentários

Java Doc de Classes

/\*\*

\* <Descrição da funcionalidade da Classe>

\* @author: <Fulano da Silva>

\*/

Java Doc de Métodos

/\*\*

\* <Descrição completa das funcionalidades do método>

\* @return <tipo do retorno do método: String, int, etc>

\* @param <parâmetro1> <Descrição do parâmetro>

\* @param <parâmetro2> <Descrição do parâmetro>

\* @exception <exceção retornada>

\*/

# Nomeação

Arquivos Java

Os arquivos Java devem usar a convenção “camel case” de nomes, começando com a primeira letra do nome maiúscula e as subsequentes minúsculas exceto a primeira letra da próxima palavra, todas as palavras devem ser escritas juntas sem espaço de separação.

Exemplo:

NomeClasse.java

Scripts SQL

Os scripts SQL devem seguir as seguintes regras:

* Prefixo com a letra “V” maiúscula
* Número da versão “1”
* Separador usando do traços “\_\_”
* Descrição com traço separando as palavras
* Sufixo “.sql”

Exemplo:

V1.0.0\_\_init.sql

# Declaração

1. Padrão de indentação de declaração de métodos:

someMethod(int anArg, Object anotherArg, String yetAnotherArg,

Object andStillAnother) {

...

}

# Reutilização

Esta seção fornece regras e diretrizes para facilitar a reutilização.

* O retorno dos dados devem ser sempre no formato de JSON
* Utilizar os conceitos de SOLID no projeto.

# Tratamento de Erros e de Exceções

* Os controladores somente tem acesso as interfaces de serviços.
* Garantir que somente as classes de implementação de serviços faça o processamento de requisições vindas dos controladores.
* Somente classes de implentação tem acesso as classes repositórios.

# teste unitário

Testes não devem ser implementados neste momento do projeto, utilize os testes manuais para validar o funcionamento do projeto.

# Diretrizes Gerais

Utilize sempre as diretrizes e recomendações de desenvolvimento de sistemas SOLID.