

UNIVERSIDADE FEDERAL DE VIÇOSA · UFV - CAMPUS FLORESTAL

CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADOR 2024

**REFINAMENTO DA SPRINT 6**

CSU06: Escolher Texto

**v.1.0.0**

Guilherme Broedel Zorzal - 5064

Arthur Fernandes Bastos - 4679

Florestal - MG

Novembro de 2024

LISTA DE FIGURAS

[**Figura 1: Diagrama de classes antigo, referente ao CSU01 3**](#_heading=h.3znysh7)

[**Figura 2: Nova organização do diagrama de classes 4**](#_heading=h.2et92p0)

[**Figura 3: Classe ConexaoHost 7**](#_heading=h.1ksv4uv)

SUMÁRIO

[**1. ATUALIZAÇÕES E CORREÇÕES 3**](#_heading=h.1fob9te)

[**2. INTRODUÇÃO 5**](#_heading=h.3dy6vkm)

[**3. CASO DE USO 6**](#_heading=h.4d34og8)

[**4. ESTRUTURA GERAL 7**](#_heading=h.3rdcrjn)

[**5. BACK-END 7**](#_heading=h.26in1rg)

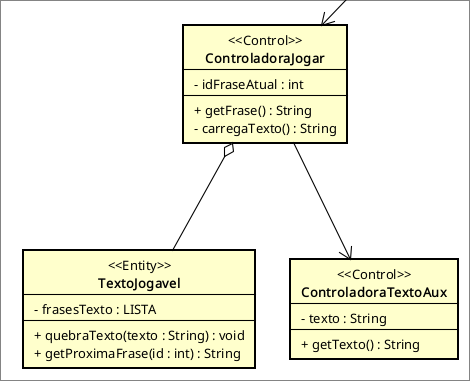
[5.1 Classe ConexaoHost 7](#_heading=h.lnxbz9)

[**6. BANCO DE DADOS 8**](#_heading=h.32hioqz)

# 1. ATUALIZAÇÕES E CORREÇÕES

Toda referência a este caso de uso e dessa sprint 06 como “selecionar texto” deve ser entendida como a funcionalidade de “Escolher texto”. Ou seja, escolher texto é a funcionalidade do back-end que escolhe o texto selecionado pelo professor.

Nessa sprint, algumas atualizações deverão ser feitas. A principal delas é a forma de funcionamento da função “carregaTexto” de ControladoraJogar. Na sprint do caso de uso Jogar, a função carrega texto era responsável por carregar o texto presente na “ControladoraTextoAux” através da função “getTexto”. Confira na imagem abaixo:

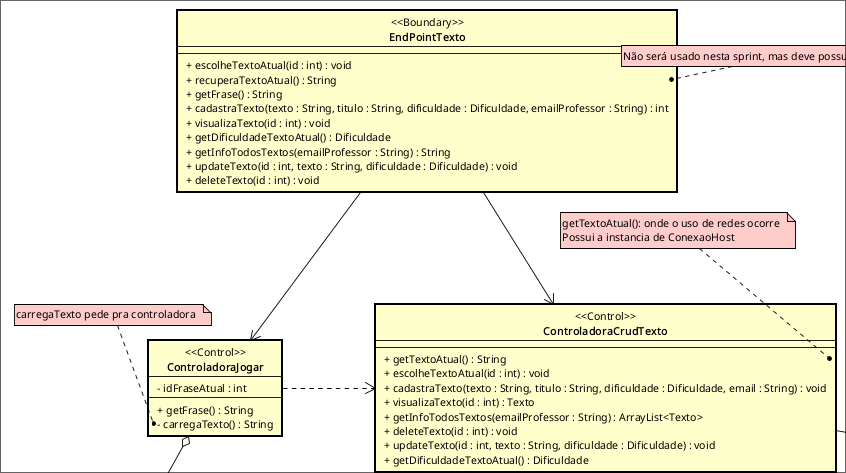


###### Figura 1: Diagrama de classes antigo, referente ao CSU01

A partir dessa sprint, o funcionamento será o seguinte: A classe ControladoraJogar terá uma referência a classe ControladoraCrudTexto presente dentro de EndPointTexto. Ou seja, ambas endPointTexto e ControladoraJogar terão uma instância do mesmo objeto. Além disso, ControladoraJogar passa a recuperar o texto não através de “getTexto”, mas sim de “getTextoAtual”, presente dentro da controladora de crud.

Lembrando que, durante a construção da controladoraJogar na sprint 3, era necessário tratar erros caso não houvesse um texto selecionado (caso o método getTexto retornasse um erro), de forma que chegasse ao front uma mensagem com essa informação. Esse fluxo de funcionamento entrará em prática nesta sprint.

As demais alterações são apenas adições de funções a classes já existentes (EndPointTexto e ControladoraCrudTexto) que serão explicadas mais adiante.



###### **Figura 2:** Nova organização do diagrama de classes

Além disso, um erro acabou passando despercebido durante a modelagem e alguns métodos terão que sofrer uma pequena alteração.

Cada professor tem seu conjunto de texto, e os textos não devem ser compartilhados entre si (essa dinâmica de como os textos serão organizados será modelado nessa sprint e pode ser conferida no fim do documento). A ideia é a seguinte: o front guardará o email do professor. Quando operações de recuperação e criação de textos forem realizadas (são as funções cadastraTexto() e getInfoTodosTextos()), o email do professor será enviado ao backend, possibilitando assim saber a quem aquele texto deve ser associado ou quais textos devem ser recuperados ao front.

Como cada texto possui um id único, a remoção e deleção de textos não precisa do uso do email do professor: o front sabe exatamente o id de cada texto e, como cada texto possui um id único, sua exclusão e atualização alterarão exatamente os textos do professor que encontra-se logado (não havendo assim a necessidade de enviar o email para essas requisições).

# 

# 2. INTRODUÇÃO

O objetivo do seguinte documento consiste em apresentar especificações relacionadas ao caso de uso abordado na quinta sprint, de acordo com o product backlog proposto com base no calendário.

Uma convenção importante com relação aos diagramas de classe: caso a representação da classe não possua nenhum método, deve-se entender que a classe precisa possuir getters e setters para seus atributos, a menos que tenham sido dadas outras instruções na descrição da classe presente no documento.

# 

# 3. CASO DE USO

O caso de uso abordado na Sprint atual ficou decidido como sendo o CSU06: Escolher texto. Sua especificação, proposta na documentação dos casos de uso em ESOF I, pode ser conferida abaixo. Note que os casos de usos sofreram poucas alterações em relação ao proposto originalmente, ou seja, o fluxo proposto inicialmente continua sendo o mesmo. Apenas algumas palavras foram atualizadas de forma a melhorar a compreensão.

Esse caso de uso é responsável por possibilitar o professor selecionar um dos textos disponíveis para uso, o texto selecionado deve ser disponibilizado para os alunos jogarem.

| CSU06: Escolher texto  **Nome:** Escolher texto.  **Ator Primário:** Professor.  **Sumário:** O professor será capaz de escolher um texto a ser utilizado pelos alunos no jogo.  **Pré-condições:**   * O usuário precisa estar autenticado.   **Fluxo Principal:**   1. O usuário seleciona a opção de escolher texto. 2. O sistema abre uma opção para que o usuário escolha qual texto será acessado pelos alunos. 3. O usuário escolhe o texto. 4. O sistema exibe o texto selecionado para os alunos.   **Pós-condições:**   * Os alunos poderão acessar o texto que foi escolhido. |
| --- |

# 4. ESTRUTURA GERAL

Como estrutura geral, nessa sprint deverá ser desenvolvido o funcionamento da requisição a partir do front-end de selecionar o texto a ser utilizado, a qual será recebida pelo back-end que terá de atualizar o texto atual e enviar o texto selecionado para ser exibido para o usuário quando necessário.

# 5. BACK-END

Na sprint atual, o back-end será responsável por implementar algumas classes e suas relações, conforme o diagrama de classes disponível no arquivo DOCX na mesma pasta. O coração da sprint atual é a classe ControladoraCrudTexto, que possui boa parte das funcionalidades relacionadas ao uso dos textos. Ela substituirá a classe ControladoraTextoAux assim que for realizado o refinamento do caso de uso SelecionarTexto (próxima sprint).

## 5.1 Classe ConexaoHost

A classe ConexaoHost não conterá nada nessa sprint. O seu único método (getTextoHost) deve retornar um erro quando chamado. Ela funcionará dessa forma visto que ela será utilizada apenas para montar o fluxo de execução das requisições de texto e deixar o ambiente de desenvolvimento preparado para receber a funcionalidade de rede.

## 

###### Figura 3: Classe ConexaoHost

O erro retornado por seu método (reforçando que nessa sprint, chamar o método deve resultar em erro) simula a conexão ao host, indicando que não foi possível realizar a conexão.

###### 

# 6. BANCO DE DADOS

Nessa sprint, o banco de dados terá o papel de realizar a união entre a tabela que guardar as informações do professor e a tabela que guarda as informações dos textos.

Cada texto deve estar associado a apenas um professor. O professor é dono de um conjunto de textos exclusivamente dele. Além disso, como já feito anteriormente, cada texto possui um id exclusivo (isso possibilita a manipulação dos textos sem a necessidade de precisar saber quem é o dono do texto: cada id referencia apenas um texto, o que já basta).

A escolha das chaves primárias e estrangeiras fica a critério dos desenvolvedores de BD, sendo que deve ficar modelado de forma que faça sentido.