Relatório Do Projeto

Fase 2

Funcionamento da Interface:

JAVAFX:

- Para selecionar vários temas na API é preciso clicar no CTRL junto do rato.
- Quando for submeter os temas como docente/utilizador Empresarial o ano do tema tem de ser o corrente no caso 2024
- É preciso o Admin fazer sempre a Atribuição do tema, depois da atribuição o aluno tem de dar logout e login de novo. Não é possível submeter proposta de tese enquanto não fizer isso.

Marcar defesa:

-Formato da data:dia-mes-ano hh:mm ex:14-04-2024 13:25

Descrição da arquitetura do projeto

O projeto foi dividido por camadas com foi dado nas aulas: user interface, business logic e data base.

Na camada de user interface usamos o padrão MVC (model, view, controller), mas de duas maneiras possíveis. Para os alunos criamos uma aplicação usando JAVAFX. Para os docentes e utilizadores empresariais usamos um site web com o springBoot e Thymeleaf.

Para a camada de business logic, criamos handlers para tratar dos casos de uso, Catálogos para obter os objetos que precisávamos e por fim usamos serviços para comunicar com os endPoints Rest chamando os Handlers.

Na base de dados tivemos as entidades criadas na fase 1 os repositórios que fazem queries, obtendo assim os objetos que queremos

Criamos ainda um WebController que trata dos pedidos do cliente web, e criamos também um ApiRestController que trata dos pedidos do cliente da aplicação.

Escolha e justificação das decisões técnicas no desenho da interface Web

Na interface web usamos o padrão MVC, organizando a estrutura desta maneira:

Controler:

Responde a pedidos Rest e trata da logica a interface em html, também trata das routes da página.

View:

Mostra as páginas para o cliente poder interagir com a logica da API.

Escolha e Justificação das Decisões Técnicas no Desenho da API REST

Estrutura dos controladores:

Dividimos os controladores em ApiRestController e WebController para separar as responsabilidades entre endpoints Rest para a aplicação e a interface web

Justificação:

- Mantém o código organizado e facilita a manutenção ao isolar a lógica Rest da logica de visualização web.
- Facilita a compreensão do código ao desenvolver, testar e fazer debug, porque cada controlador tem um conjunto bem definido de responsabilidades.

Utilização de DTOs (Data Transfer Objects):

Utilizamos DTOs para transferir dados entre a camada de aplicação e os controladores API REST.

Justificação:

- DTOs permitem encapsular os dados de maneira mais eficiente, expondo apenas o necessário e mantendo a integridade das entidades. No nosso projeto transformamos objetos em ids facilitando o envio do objeto. Por exemplo o Aluno tinha um objeto Tema para tema atribuído, enquanto o AlunoDTO tinha o id para tema atribuído.

Enpoints Rest:

Adotamos convenções Rest para a definição dos enpoints, como o uso de métodos HTTP adequados (GET, POST, PATCH) e a organização de URLs de forma hierárquica e semântica.

Justificação:

- Seguir as convenções facilita a compreensão e a utilização da API por outros desenvolvedores
- Foi dado nas aulas T e TPs

Escolha e Justificação das Decisões Técnicas no Desenho da Interface JavaFX

Na interface JavaFX também usamos o padrão MVC, por isso a nossa estrutura da aplicação em javaFX foi feita da seguinte forma:

Controlers:

Nesta pasta temos toda a lógica da interface desde o que fazer quando clico num botão até enviar pedidos ao servidor e alterar o tipo de página a ser mostrada.

View:

Nesta pasta temos a formulação do que será mostrado para o cliente com alguns botões lables e texto.

Model:

Nesta pasta temos uma classe model que é responsável por gerir e armazenar dados da aplicação especificamente os dados relacionados ao AlunoDTO.