Relatório de Projeto: Construção de Sistemas de Software

Projeto: ThesisMan - Gestão de Teses de Mestrado

Autores: Vasco Baldé (58174), José Martins (58228), Maria Silva (58243)

---

1. Descrição da Arquitetura do Projeto

O projeto ThesisMan está organizado em duas pastas principais: uma contendo o código que executa o servidor PostgreSQL e outra contendo o código JavaFX para a interface do utilizador. A estrutura de diretórios do projeto é a seguinte:

Estrutura de Diretórios

- Servidor PostgreSQL
    - pt.ul.fc.css.thesisman
        - *appConfig*: Configurações da aplicação.
        - *business.services*: Contém objetos DTO enviados pela API.
        - *business.services.exceptions*: Exceções utilizadas nos serviços.
        - *controller*: Implementação da API REST.
        - *datatypes*: Tipos de dados auxiliares.
        - *entities*: Entidades manipuladas pelo projeto, muitas com versões DTO.
        - *handlers*: Classes auxiliares para processamento de pedidos.
        - *repositories*: Repositórios JPA para as entidades.
        - *service*: Serviços chamados pelos controladores para processar APIs e validar dados.
- Cliente JavaFX

- pt.ul.fc.css.javafxclient.presentation.model: Modelos de Aluno e Mestrado recebidos da API.

2. Escolha e Justificação das Decisões Técnicas na Arquitetura da Aplicação

Optámos por não enviar diretamente as entidades tal como são armazenadas nos repositórios através da API. Em vez disso, utilizamos DTOs (Data Transfer Objects). Esta abordagem oferece várias vantagens:

- *Segurança*: Evita a exposição direta da estrutura interna da base de dados.
- *Validação*: Facilita a validação dos dados antes de serem processados ou armazenados.
- *Flexibilidade*: Permite ajustar a estrutura das respostas da API sem alterar o modelo de dados subjacente.

3. Escolha e Justificação das Decisões Técnicas no Desenho da Interface Web

Para o desenho da interface web, foram tomadas as seguintes decisões:

- Tecnologia:Utilizamos a tecnologia Server-Side Rendering, como pedido no enunciado.
- UX/UI: Desenhamos uma UI bastante básica, de modo a facilitar o utilizador.

4. Escolha e Justificação das Decisões Técnicas no Desenho da API REST

4.1. Estrutura RESTful

Adotamos uma estrutura RESTful para garantir uma API intuitiva e fácil de usar, com endpoints específicos para cada recurso e métodos HTTP adequados (GET, POST, PUT).

4.2. Segregação de Funções

Dividimos a lógica em diferentes controladores para organização e manutenção:

- `RestAlunoController`: Autenticação e gestão de alunos.
- `RestDocenteController`: Registro e gestão de docentes.
- `RestUtilizadorEmpresarialController`: Registro e login de utilizadores empresariais.

4.3. Uso de DTOs (Data Transfer Objects)

Utilizamos DTOs para a comunicação entre cliente e servidor, protegendo a estrutura interna da base de dados e facilitando a adaptação da API.

4.4. Tratamento de Exceções

Implementámos um tratamento robusto de exceções:

- Erros de autenticação: `HttpStatus.UNAUTHORIZED` ou `HttpStatus.FORBIDDEN`.
- Campos obrigatórios em falta: `HttpStatus.BAD_REQUEST`.
- Exceções internas: `HttpStatus.INTERNAL_SERVER_ERROR`.

4.5. Autenticação e Autorização

Utilizamos JWT (JSON Web Tokens) para autenticação:

- Login: Gera um token para o utilizador autenticado.
- Logout: Invalida o token.
- Verificação de Token: Protege endpoints sensíveis

5. Integração do Cliente JavaFX com a API REST

Estrutura do Cliente JavaFX

O cliente JavaFX do projeto ThesisMan foi desenvolvido para interagir com a API REST, permitindo a autenticação, visualização de temas e submissão de documentos. A estrutura do cliente é composta por diferentes controladores e cenas.

Autenticação do Utilizador

- LoginController: Controla a lógica de autenticação. Quando o utilizador insere as suas credenciais e clica no botão de login, é feita uma chamada à API para verificar as credenciais. Se o login for bem-sucedido, a aplicação redireciona para o menu principal.

Gestão de Temas

- *TemasController*: Carrega e exibe os temas disponíveis. A interação com a API ocorre para buscar temas e enviar candidaturas. O utilizador pode visualizar os temas compatíveis e candidatar-se a eles, utilizando botões na interface.

Submissão de Documentos

- *SubmissionController*: Responsável pela submissão de documentos de proposta e tese final. Realiza chamadas à API para enviar os documentos selecionados pelo utilizador. A interface permite a seleção dos ficheiros e a submissão através de botões.

Interface do Utilizador

- Ficheiros FXML: Os ficheiros FXML definem a interface do utilizador:
  - `login.fxml`: Tela de login com campos de texto e botão de login.
  - `menu.fxml`: Tela de gestão de temas e submissão de documentos. Inclui botões para carregar temas, candidatar-se, submeter documentos e fazer logout.

Notas: Devido à falta tempo e excesso que entregas acumuladas que levaram a falta de tempo necessário para a produção do projeto, não conseguimos colocar todas as funcionalidades operacionais