Trabalho Final - Desenvolvimento de um Sistema REST API com Spring Boot em uma Arquitetura Monolítica

> **Disciplina**: Projeto Backend Monolítico com ORM
> **Prof**.: Camilo Barreto
> **Período**: 3°
> **Curso**: Tec. Sistemas para Internet
> **IFTM - Campus Uberlândia Centro**

1. Introdução

Neste trabalho, vamos explorar o desenvolvimento de um sistema **REST API** utilizando o framework **Spring Boot** e adotando uma **arquitetura monolítica**. O objetivo é fornecer aos alunos uma oportunidade de aprender e aplicar conceitos fundamentais de desenvolvimento web, como criação de endpoints REST, persistência de dados e gerenciamento de dependências com o Spring Boot.

O escopo do trabalho envolve a criação de uma aplicação Web Service que expõe uma API REST para gerenciar determinada funcionalidade, como um sistema de gerenciamento de tarefas, um blog ou um sistema de reserva de ingressos. Os alunos serão responsáveis por definir os requisitos principais do sistema, identificando as entidades envolvidas, os recursos necessários e as operações suportadas pela API.

Durante o desenvolvimento, os alunos irão se aprofundar em diversos aspectos essenciais do desenvolvimento web. Eles irão implementar a persistência de dados utilizando um banco de dados relacional, explorando os relacionamentos entre as entidades, como One-to-One, One-to-Many e Many-to-Many. Além disso, serão desafiados a criar queries customizadas para atender a necessidades específicas do sistema.

A segurança também será um ponto importante do trabalho. Os alunos irão implementar autenticação e autorização utilizando o Spring Security, garantindo que apenas usuários autenticados tenham acesso aos recursos adequados. Será utilizado o conceito de JSON Web Tokens (JWT) para o gerenciamento de sessões e controle de acesso.

Outro aspecto fundamental a ser explorado é a documentação da API. Os alunos irão utilizar o OpenAPI 3 e o Swagger para documentar de forma clara e precisa os endpoints, parâmetros, respostas e outras informações relevantes. Isso facilitará o entendimento da API por parte de outros desenvolvedores e usuários.

Ao longo do trabalho, os alunos deverão adotar boas práticas de desenvolvimento, como o tratamento adequado de exceções, o uso de Content Negotiation para suportar diferentes formatos de resposta (JSON, XML e YAML) e a implementação do conceito de HATEOAS para fornecer links de navegação nos recursos da API.

Ao finalizar o trabalho, os alunos terão adquirido habilidades práticas em desenvolvimento de APIs REST com Spring Boot em uma arquitetura monolítica. Eles terão conhecimento sobre persistência de dados, relacionamentos entre entidades, tratamento de exceções, segurança com Spring Security e JWT, documentação com OpenAPI 3 e Swagger, entre outros conceitos essenciais para o desenvolvimento de sistemas web robustos e escaláveis.

Informações Sobre a Equipe:

- Trabalho deverá ser realizado com no máximo DOIS alunos;
- As tarefas devem ser divididas igualmente aos integrantes (caso duplas);

⚠️ Prazos Gerais:

Data de publicação: 23/05/2023

Data de entrega: 03/07/2023

Valor: 60 pontos;

2. Etapas do Desenvolvimento do Projeto

Etapa 1 - Definir o Escopo do Projeto:

Nesta etapa, os alunos devem definir qual será a aplicabilidade do sistema a ser desenvolvido. Eles devem escolher um domínio específico, como um sistema de gerenciamento de tarefas ou um sistema de ordem de serviço.

Etapa 2 - Especificar o Sistema:

Nesta etapa, os alunos devem especificar as funcionalidades do sistema. Eles devem descrever o que o sistema será capaz de fazer por meio dos Requisitos Funcionais. Por exemplo, para um sistema de ordem de serviço, um requisito funcional pode ser "RF1 - O sistema deve ser capaz de registrar uma nova ordem de serviço". Os alunos devem criar uma lista de requisitos do sistema. Exemplo:

- RF1 - O sistema deverá fornecer um endpoint para buscar elementos;
- RF2 - O Sistema deverá registrar um elemento;
- RF3 - …

Para mais informações de como criar os Requisitos Funcionais:

- https://www.mestresdaweb.com.br/tecnologias/requisitos-funcionais-e-nao-funcionais-o-que-sao
- https://www.linkedin.com/pulse/requisitos-funcionais-e-não-vânia-porto-da-silva/?originalSubdomain=pt

---

Etapa 3 - Configuração do Ambiente de Desenvolvimento:

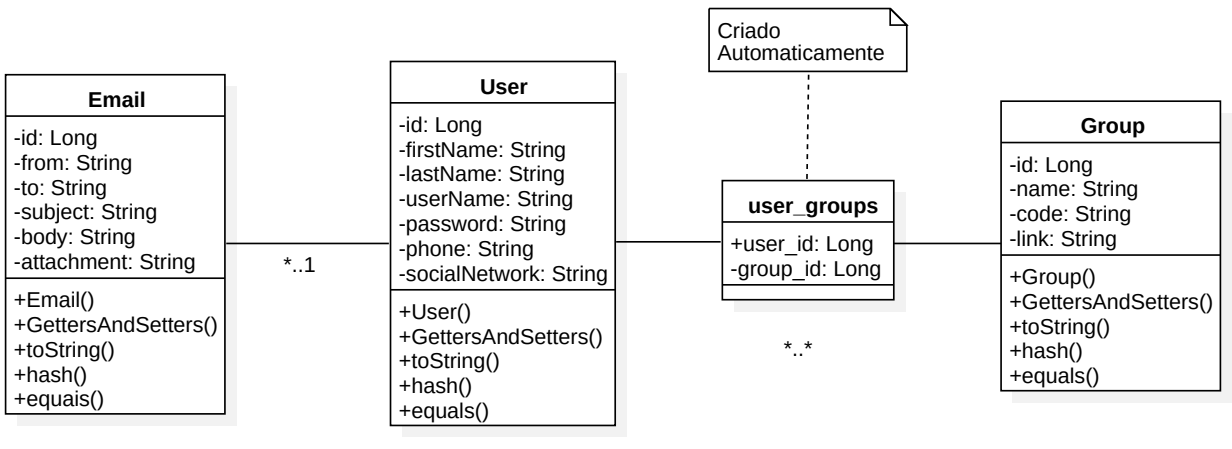
Os alunos devem configurar o ambiente de desenvolvimento para trabalhar com o framework Spring Boot, banco de dados, Postman (para testar as APIs), um navegador (para acessar a documentação) e o GitHub (para versionamento do código e gerenciamento do projeto).

Etapa 4 - Projeto UML do Relacionamento das Entidades do Banco de Dados:

Nesta etapa, os alunos devem definir e projetar as entidades do banco de dados e seus relacionamentos. Eles devem criar um diagrama UML que represente as entidades e os relacionamentos, garantindo que haja relacionamentos:

- One-to-One;
- One-to-Many;
- Many-to-Many.

Um exemplo de diagrama UML é apresentado abaixo:



Etapa 5 - Criar um Repositório para o Projeto no GitHub:

Os alunos devem criar um repositório no GitHub para armazenar o projeto. O professor deve ser adicionado como colaborador do projeto. Os alunos devem criar um "Projeto" no GitHub para gerenciar o desenvolvimento do trabalho. Eles devem criar tarefas e atribuí-las aos integrantes, definindo prazos de entrega por meio de milestones.

Usuário Github do professor: **CamiloJr**

- https://github.com/CamiloJr

Links para ajuda:

- https://docs.github.com/pt/issues/organizing-your-work-with-project-boards/managing-project-boards/creating-a-project-board
- https://docs.github.com/pt/issues/planning-and-tracking-with-projects/creating-projects/creating-a-project
- Para criar Milestones: https://docs.github.com/pt/issues/using-labels-and-milestones-to-track-work/creating-and-editing-milestones-for-issues-and-pull-requests
- Para quem nunca usou o Github: https://www.youtube.com/watch?v=_hZf1teRFNg

---

Etapa 6 - Desenvolvimento e Versionamento com Git:

Nesta etapa, a dupla deverá iniciar a implementação do projeto. Após definir os requisitos do sistema, as entidades do banco de dados e seus relacionamentos, então a implementação deverá ser iniciada. O desenvolvimento deverá ser dividido entre a dupla e versionada no Github, isto é, cada um deverá trabalhar em uma parte do projeto e versionar no Github com contas diferentes. Deveram seguir as tarefas atribuídas aos integrantes através do sistema de projetos do Github. É obrigatório que o desenvolvimento do sistema contemple os seguintes tópicos:

- [ ] Arquitetura Monolítica;
- [ ] Persistência com Banco de Dados;
- [ ] Relacionamento entre Entidades de One-to-One, One-to-Many e Many-to-Many;
- [ ] Queries customizadas;
- [ ] Tratamento de exceções customizadas;
- [ ] Content Negotiation (JSON, XML e YAML);
- [ ] HATEOAS;
- [ ] Endpoints CRUD para cada entidade;
- [ ] Segurança com Spring Security e JWT;
- [ ] Documentação com OpenAPI 3 e Swagger;

Etapa 7 - Apresentação da Documentação do Projeto e dos Endpoints do Swagger:

Nesta etapa os alunos deveram apresentar a documentação:

- Requisitos Funcionais do sistema;
- Endpoints com Swagger e Postman;

A documentação deverá ser atribuída no READ-ME do Github. Criem um READ-ME de fácil compreensão e de forma clara.

Formas de Avaliação:

O processo de avaliação do trabalho será realizado com base nas etapas descritas anteriormente. Cada etapa do desenvolvimento do sistema será avaliada de maneira específica, levando em consideração os objetivos e as datas de entregas especificadas e a qualidade do projeto entregue.

Na Etapa 1 (03 pontos): A avaliação se concentrará na capacidade dos alunos em identificar e delimitar claramente a aplicabilidade do sistema a ser desenvolvido e a entrega dentro do prazo;

Na Etapa 2 (07 pontos): A avaliação considerará a habilidade dos alunos em descrever as funcionalidades do sistema por meio dos Requisitos Funcionais e a entrega dentro do prazo;

Na Etapa 4 (05 pontos): A avaliação considerará a capacidade dos alunos em projetar corretamente as entidades do banco de dados e seus relacionamentos através de um diagrama UML e a entrega dentro do prazo;

Na Etapa 5 (05 pontos): A avaliação pode considerar a habilidade dos alunos em criar corretamente um repositório no GitHub e gerenciar o projeto por meio de tarefas e milestones.

Na Etapa 6 (35 pontos): A avaliação considerará a capacidade dos alunos em implementar corretamente as funcionalidades do sistema, levando em consideração os requisitos especificados anteriormente. Os alunos devem ser capazes de dividir o trabalho entre si, versionar o código corretamente no GitHub e seguir as tarefas atribuídas a cada integrante.

- **(5 pontos)** Relacionamento entre Entidades de One-to-One, One-to-Many e Many-to-Many;
- **(3 pontos)** Queries customizadas;
- **(4 pontos)** Tratamento de exceções customizadas;
- **(4 pontos)** Content Negotiation (JSON, XML e YAML);
- **(3 pontos)** HATEOAS;
- **(5 pontos)** Endpoints CRUD para cada entidade;
- **(4 pontos)** Segurança com Spring Security e JWT;
- **(3 pontos)** Documentação com OpenAPI 3 e Swagger;
- **(4 pontos)** Versionamento no Github;

Na Etapa 7 (05 pontos): A avaliação considerará a capacidade dos alunos em apresentar de forma clara e organizada a documentação do projeto. Isso inclui os Requisitos Funcionais do sistema, a documentação dos endpoints da API utilizando o Swagger e a entrega dentro do prazo;

Cronograma de Entregas:

Fiquem atentos ao cronograma de entregas.

> 💡 A cada dia de atraso será descontado 2 pontos da etapa, podendo ser zerada se não entregar em tempo hábil.

Etapa:	Data de Entrega:	Pontuação:
1	23/05/2023	03
2	30/05/2023	07
3	30/05/2023	00
4	06/06/2023	05
5	06/06/2023	05
6	03/07/2023	35
7	03/07/2023	05

Bom trabalho!