

Caros estudantes,

Conforme o método de avaliação apresentado no plano de ensino no início do semestre, a NP2 da disciplina tópicos especiais em programação web se dará a partir de um projeto que englobe os conteúdos já abordados na disciplina. O projeto poderá ser realizado por equipes de até 03 alunos.

Objetivo: Construir o backend de uma aplicação (API REST) que atenda os requisitos e regras abaixo descritos.

Requisitos:

- Definição e validação dos requisitos de um software de biblioteca virtual pública;
- Definição de cronograma de entregas por membro;
- Desenvolver rotas para 05 CRUDs completos (cadastrar, editar, pesquisar todos, pesquisar por ID e excluir);
- Desenvolver pelo menos 01 funcionalidade (que não envolva CRUD) no contexto especificado, sendo avaliado pela complexidade do desenvolvimento;
- As entidades deverão possuir ao menos 04 atributos (sem contar o ID), podendo diminuir mediante justificativa do contexto;
- Todas as entidades deverão possuir as camadas e padrões de projeto abordados em sala (Model, DTO, Repository, Service e Controller);
- As possíveis exceções/erros deverão ser devidamente tratados e retornados de uma forma amigável na rota (com seus respectivos Status Code);
- Todas as operações de CRUD das entidades deverão ser executadas em uma base de dados;
- Todo o DDL das tabelas das entidades deverão ser executados via Migration;
- Todas as rotas da API deverão ser completamente documentadas (Swagger);
- Repositório (github) com o projeto;
- Deverá ser submetido um .zip contendo o Projeto construído (código fonte) + nome dos membros da equipe + link do repositório github.

Regras:

- A linguagem de programação utilizada fica a critério do estudante (contanto que aborde todos os tópicos dos requisitos);
- As funcionalidades desenvolvidas deverão estar no contexto especificado (Biblioteca Virtual Pública)
- Todos os CRUDs deverão estar em um contexto semântico com o tema;
- Será obrigatório registros de commit de ambos os membros da dupla;
- Todos deverão apresentar o projeto (5 a 8 min de apresentação);
- Na apresentação deve ser especificado o desenvolvido de cada membro (conforme o cronograma apresentado inicialmente);
- Se não entregar o projeto, a nota definida para a equipe será 0;
- Parte da nota será individual (40%), a depender das entregas realizadas.
- Pode remover membros até dia 14/04;

- Caso a equipe/aluno falte, poderá realizar uma segunda chamada (da apresentação);
- A apresentação tem nota individual, faltando, terá nota 0 atribuída;
- Todos os alunos deverão responder 02 perguntas após a apresentação, elaborada pelos próprios colegas ou pelo professor;
- Não será considerado como CRUD, a entidade de usuário construída nas aulas
- A apresentação do projeto deverá ser realizada por ambos os membros da dupla

Método de Avaliação:

- 70% Projeto (40% individual)
- 30% Apresentação

considere os seguintes requisitos do trabalho da disciplina de backend do curso de análise e desenvolvimento de sistemas sobre o desenvolvimento de um software de biblioteca virtual pública:

Criar no mínimo 5 atributos;

Desenvolver rotas para 05 CRUDs completos (cadastrar, editar, pesquisar todos, pesquisar por ID e excluir);

Desenvolver pelo menos 01 funcionalidade (que não envolva CRUD) no contexto especificado, sendo avaliado pela complexidade do desenvolvimento;

As entidades deverão possuir ao menos 04 atributos (sem contar o ID), podendo diminuir mediante justificativa do contexto;

Todas as entidades deverão possuir as camadas e padrões de projeto abordados em sala (Model, DTO, Repository, Service e Controller);

As possíveis exceções/erros deverão ser devidamente tratados e retornados de uma forma amigável na rota (com seus respectivos Status Code);

Todas as operações de CRUD das entidades deverão ser executadas em uma base de dados;

Todo o DDL das tabelas das entidades deverão ser executados via Migration;

Todas as rotas da API deverão ser completamente documentadas (Swagger);

Upload e armazenamento dos PDFs dos livros (incluindo rota para recebê-los);

me diga uma divisão dessas tarefas entre os 3 membros de uma equipe de maneira que o trabalho fique dividido de uma maneira justa (todos os integrantes com mesmo nível de dificuldade e de tempo gasto). Considere que cada CRUD é uma atividade, ou seja serão 5 atividades de CRUD que podem ser divididas entre os membros, ou ficar com a mesma pessoa, se fizer sentido.

O ChatGPT disse:

Claro! Vamos organizar uma divisão justa das tarefas considerando:

- **5 CRUDs completos** (cada um com rotas de cadastrar, editar, listar todos, listar por ID e deletar),
- **1 funcionalidade extra que não envolva CRUD**,
- **Upload e armazenamento dos PDFs dos livros (incluindo rota para recebê-los)** como **uma atividade separada** (atividade técnica e específica),
- Estrutura completa com **Model, DTO, Repository, Service e Controller** para todas as entidades,
- **Tratamento de erros e exceções**,
- **Migrations**,
- **Documentação Swagger**.



Proposta de divisão entre 3 membros

Membro 1 – Foco em CRUDs (2 CRUDs + parte comum a todos)

Atividades:

- Desenvolver **2 CRUDs completos** (com Model, DTO, Repository, Service, Controller).
- Criar as **Migrations** das suas entidades.
- **Tratar erros** (HTTP Status, mensagens amigáveis) nas rotas que criar.
- Participar da **documentação Swagger** das rotas que criou.
- Participar da **revisão em pares** da estrutura das camadas.

Membro 2 – Foco em CRUDs (3 CRUDs + parte comum a todos)

Atividades:

- Desenvolver **3 CRUDs completos**.
- Criar as **Migrations** das suas entidades.
- **Tratar erros** nas suas rotas.

- Participar da **documentação Swagger**.
- Revisar o código do colega (revisão cruzada é sempre bem-vinda).

Como este membro fará um CRUD a mais, pode pegar entidades mais simples (com menos regras de negócio), para balancear.

Membro 3 – Foco na funcionalidade extra e upload dos PDFs

Atividades:

- Desenvolver a **funcionalidade extra** (não-CRUD). Ex: gerar um relatório de livros mais alugados, ou verificar disponibilidade de livros para aluguel por período.
- Implementar o **upload e armazenamento dos PDFs** em serviço como o Amazon S3:
 - Criar a **rota de upload de arquivo**.
 - Gerar e salvar o link de acesso ao PDF.
- Tratar **exceções relacionadas a arquivos/S3**.
- **Documentar via Swagger** as rotas implementadas.
- Auxiliar na estrutura comum e revisão de código.

Justificativa da divisão

- O **upload de PDF + funcionalidade extra** envolve conhecimento técnico adicional (como multipart/form-data, integração com serviço externo), o que balanceia com quem está fazendo mais CRUDs.
- As tarefas são divididas em blocos grandes, mas dentro deles, as camadas estão distribuídas igualmente.
- Cada um participa ativamente da documentação, garantindo que todos conheçam bem o projeto.
- O tempo e complexidade estimados ficam nivelados.

Sugestão de 5 Entidades para os CRUDs

1. Usuário

Representa quem acessa a biblioteca virtual (público geral).

Atributos:

- nome
- email
- senha (hash)
- data de nascimento

📌 CRUD completo: cadastrar, editar, listar, buscar por ID, excluir

2. Livro

Representa os livros cadastrados, com seus metadados e link para o PDF.

Atributos:

- título
- autor
- ano de publicação
- categoria
- url_pdf (link no S3)

📌 CRUD completo

📌 Aqui entra a **atividade de upload de PDF** (Membro 3 vai fazer a rota de upload e armazenar no S3, os outros membros usam o link gerado no CRUD do livro)

3. Categoria

Categorias dos livros (Ficção, História, Ciência, etc.)

Atributos:

- nome
- descrição
- público alvo
- código da categoria

📌 CRUD completo

📌 Pode ser uma entidade mais simples (para o membro com 3 CRUDs)

4. Aluguel

Representa um livro sendo alugado por um usuário.

Atributos:

- id_livro
- id_usuario
- data_aluguel
- data_devolucao
- status (ativo, devolvido, atrasado)

📌 CRUD completo

5. Autor

Informações sobre autores (em vez de só nome no livro).

Atributos:

- nome completo
- nacionalidade
- data de nascimento
- biografia

📌 CRUD completo

📌 Também pode ser um CRUD mais simples

✨ Funcionalidade Extra (Não-CRUD)

Aqui vão algumas ideias com níveis variados de complexidade:

♦ Opção 1 (Simples):

Buscar livros disponíveis por categoria

- Exemplo: `GET /livros/disponiveis?categoria=Ficção`
- Mostra livros daquela categoria que não estão alugados no momento.

♦ Opção 2 (Intermediária):

Relatório de livros mais alugados

- Rota: `GET /relatorios/livros-mais-alugados`
- Retorna os livros ordenados por número de vezes alugado.

♦ Opção 3 (Avançada):

Verificar disponibilidade por período

- O usuário informa uma data e o sistema responde se o livro está livre naquele intervalo.

📌 A opção 2 é bem interessante e dá um charme a mais ao projeto, além de ter lógica suficiente para ser considerada uma boa funcionalidade extra.

Conclusão (Resumo da divisão com entidades)

Membro	Tarefas
1	CRUD Usuário + CRUD Autor + documentação + migrations + erros
2	CRUD Livro + CRUD Categoria + CRUD Aluguel + documentação + migrations + erros
3	Funcionalidade extra (relatório) + Upload de PDF + rota de upload + integração S3 + Swagger + tratamento de erros específicos