



## Projeto Collegialis

### Introdução

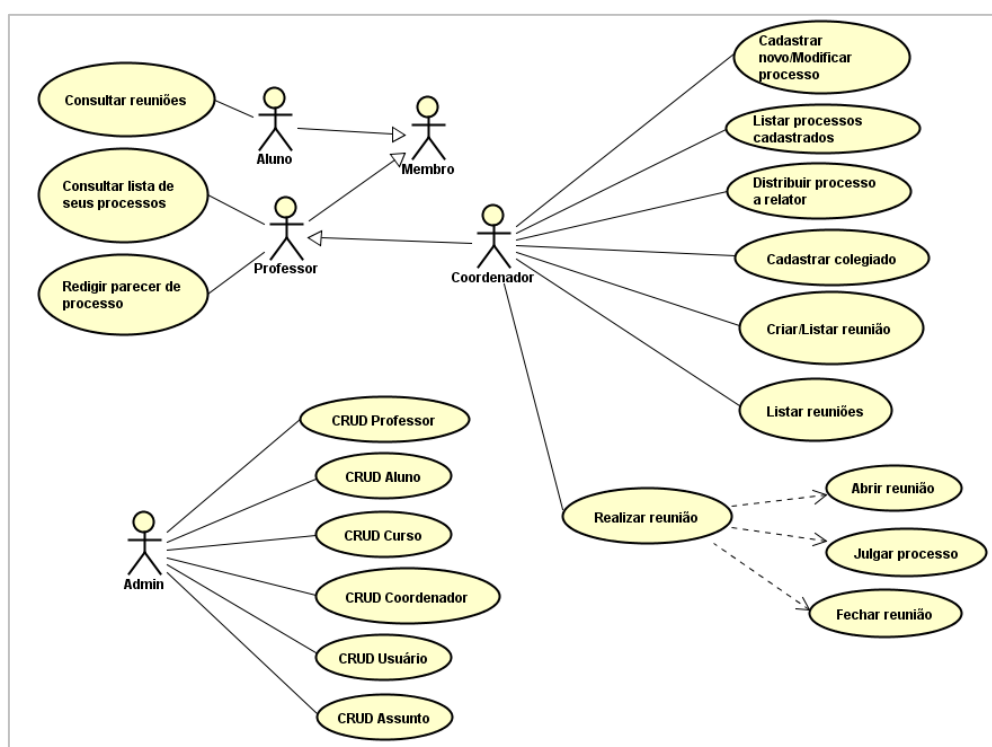
Deseja-se um sistema que controle o fluxo de julgamento de processos para um colegiado de professores. O colegiado é um grupo de professores que se reúnem para julgar processos criados por alunos sobre assuntos acadêmicos. O sistema deve possuir diversos perfis de usuários, cada um com acesso a funcionalidades específicas.

Um aluno abre um processo fazendo uma determinada solicitação acadêmica (reabertura de matrícula ou dilatação do prazo para término do curso, por exemplo). O coordenador do curso recebe este processo e o encaminha para um professor julgá-lo se procedente ou não (professor relator). O coordenador marca reuniões deste colegiado se reunir e julgar os processos que foram colocados na pauta desta reunião. O relator apresenta seu voto (de acordo ou não com o pleito do aluno) e os demais membros do colegiado votam se concordam ou não com o relator. A maioria dos votos define o resultado da análise do processo.

Ao aluno cabem as funcionalidades de criar processos e ter acesso aos andamentos destes processos (se já foi distribuído para um relator, se está na pauta de alguma reunião etc.). Ao coordenador cabem as funcionalidades de distribuição de processos, acompanhamento do estado de todos os processos (se já foi julgado pelo relator, se está em pauta etc.), de criar uma reunião e definir sua pauta (processos que serão julgados na reunião) e data de acontecimento, de ver as pautas já criadas e de presidir a reunião (consultar a pauta e ir julgando cada processo). Ao professor membro do colegiado cabem as funcionalidades de dar seu voto a um processo (informar se concorda ou não com o pedido do aluno), consultar os processos que foram delegados para si pelo coordenador etc.

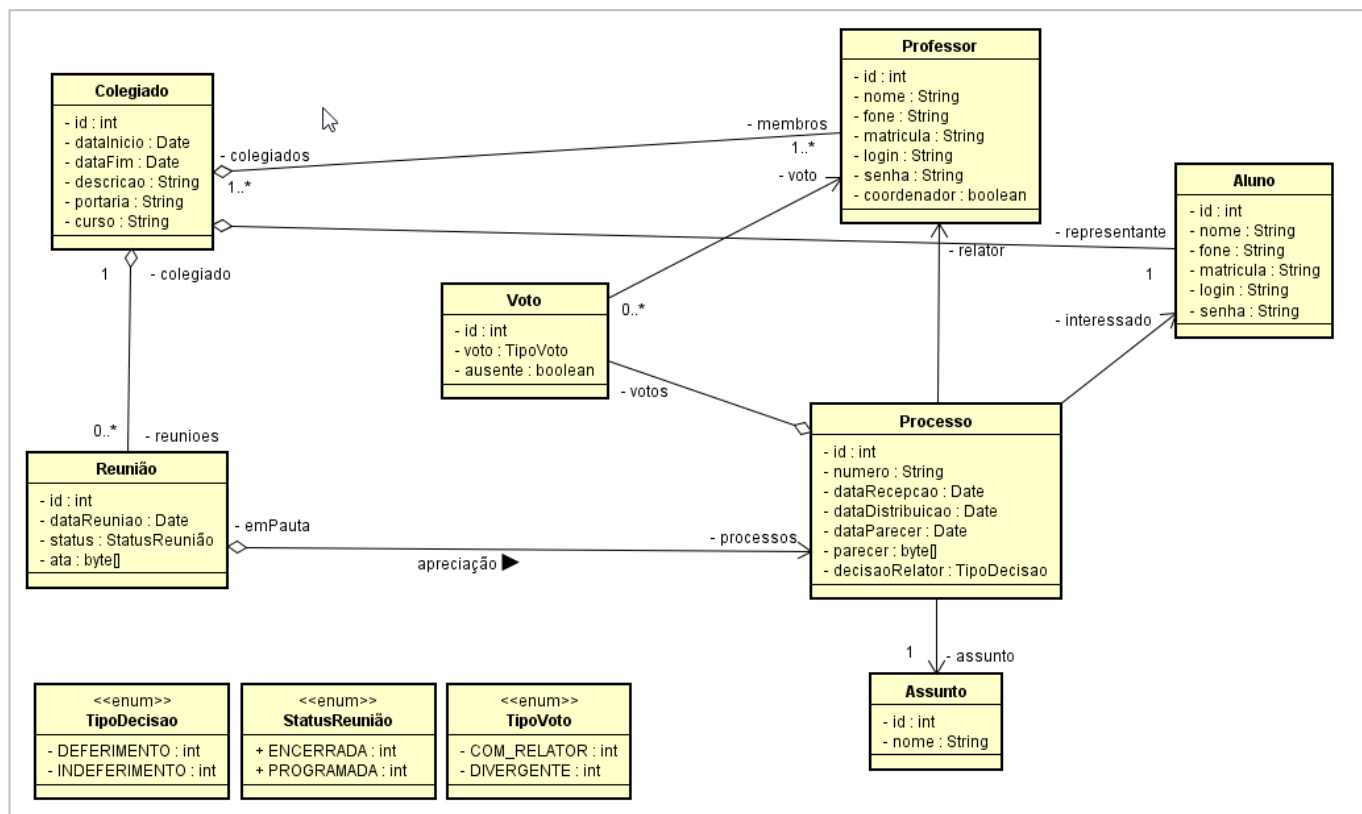
### Diagramas

#### Diagrama de Casos de Usos

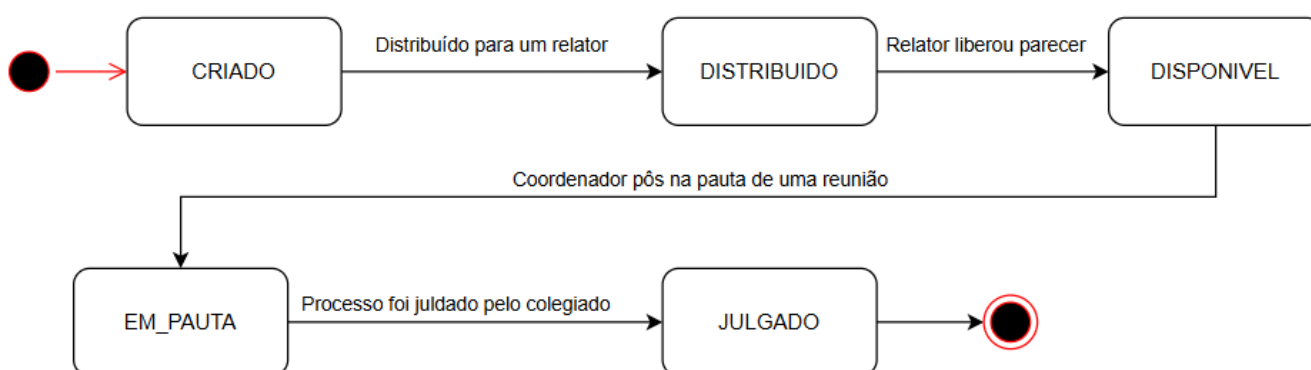




## Diagrama de Classe



## Diagrama de Estados



Estes diagramas são apenas uma sugestão, um ponto de partida e podem e devem ser melhorados e refinados.



## Protótipos de algumas telas

### Planejamento de Reuniões

**Nova** Status: **PLANEJADAS**

Reunião	Data	#processos	Status	Operações	Excluir
Reunião 05/2016	08/06/2016	5	PLAN	PDF	X
Reunião 06/2016	12/07/2016	4	PLAN	PDF	X

Emite PDF da ata

#### Nova reunião

Reunião:  /   
Data:

Observações:  
Reunião adiada em 12/07 para 19/07.

Processos:

Nº	Requisitante	Assunto	Relator	Excluir
36237464/22	José Silva	Matricula em discipl	Frederico Pereira	<input type="checkbox"/>
43984373/32	Mariana Mendes	Reabertura matrícu	Jaildo Pequeno	<input type="checkbox"/>
34498439/95	Fernando Matos	Reabertura matrícu	Valéria Cavalcanti	<input type="checkbox"/>

**Salvar**

Tela de planejamento de uma sessão do colegiado

### Reunião 0004

Data:

**Pauta** **Processo em apreciação**

233423400435/18 <  
491179334359/26 \*  
114984566668/04 \*  
481102342474/36  
241183435476/12

**Encerrar reunião**

Nº:  Solicitante:

Assunto:

Relator:  Decisão:  Parecer:

Votos

Frederico Costa

Com o relator

Divergente

Ausente

Marcos Varandas

Com o relator

Divergente

Ausente

Jaildo Pequeno

Com o relator

Divergente

Ausente

Jonathan Silva

Com o relator

Divergente

Ausente

**Concluir**

**Retirar de pauta**

Tela de condução de uma sessão de julgamento



## **Requisitos Funcionais**

Além do modelo de classes, os seguintes requisitos funcionais foram colhidos:

- REQFUNC 1. O aluno cadastra um novo processo, informando o assunto e um texto com seu requerimento.
- REQFUNC 2. O aluno consulta o estado de todos os seus processos, podendo filtrá-los pelo status ou assunto e ordená-los pela data de criação.
- REQFUNC 3. O professor consulta todos os processos que lhe foram designados pelo coordenador.
- REQFUNC 4. O professor consulta todas as reuniões do colegiado, podendo filtrar pelo status dela (finalizada ou agendada).
- REQFUNC 5. O professor vota pelo deferimento ou indeferimento de um processo, podendo redigir um texto com sua justificativa de voto.
- REQFUNC 6. O professor consulta reuniões agendadas para o colegiado nas quais foi escalado para participar.
- REQFUNC 7. O coordenador consulta todos os processos do colegiado, podendo filtrá-los por seu status, pelo aluno interessado ou pelo professor relator (caso já tenha sido distribuído).
- REQFUNC 8. O coordenador pode distribuir um processo para um professor membro do colegiado, o qual será designado relator do processo.
- REQFUNC 9. O coordenador pode criar uma sessão do colegiado e definir sua data, sua pauta (processos que serão julgados pelo colegiado) e os membros do colegiado que participarão dela.
- REQFUNC 10. O coordenador deve iniciar uma sessão de julgamento. Apenas uma sessão pode estar iniciada por vez.
- REQFUNC 11. O coordenador deve apregoar<sup>1</sup> cada processo da pauta da sessão, indicando para o mesmo como cada membro do colegiado votou (ver tela de condução da sessão). Ao término do apregoamento, o sistema deve calcular automaticamente o resultado do processo (deferido ou indeferido). Se o relator votou pelo deferimento (ou indeferimento) e todos ou a maioria dos membros do colegiado votaram com ele, o processo é considerado deferido (ou indeferido). Caso a maioria tenha votado divergente do relator, o processo terá como resultado o julgamento contrário ao do relator.
- REQFUNC 12. O coordenador deve finalizar uma sessão, caso em que, nenhuma informação acerca dos julgamentos dos processos poderá mais ser alterada.
- REQFUNC 13. O administrador deve realizar CRUDs para colegiados (conjunto de professores que o compõem).
- REQFUNC 14. O administrador deve realizar CRUDs para alunos, professores e coordenadores.
- REQFUNC 15. O administrador deve realizar CRUDs para assuntos de processos.
- REQFUNC 16. Aluno faz upload (PDF) de texto do requerimento, durante ou após criação do processo. O processo não pode estar distribuído ainda.

## **Requisitos não funcionais**

---

<sup>1</sup> Apregoar é designar, dentre os processos da pauta, aquele que está em julgamento no momento.



- REQNAOFUNC 1. : Utilizar o Spring Boot na implementação.  
REQNAOFUNC 2. : Utilizar framework de CSS (Bootstrap, Tailwind etc).  
REQNAOFUNC 3. : Qualquer banco de dados relacional.  
REQNAOFUNC 4. : Utilizar o Hibernate como provider de persistência.  
REQNAOFUNC 5. : Utilizar validação e campos de mensagens para mostrar erros em todas os formulários.  
REQNAOFUNC 6. : Utilizar o padrão Post\_Redirect\_Get.  
REQNAOFUNC 7. : Utilizar layouts e fragmentos para os templates Tymeleaf.  
REQNAOFUNC 8. : Utilizar mecanismos de autenticação e autorização do Spring Security.  
REQNAOFUNC 9. : Utilizou paginação em tabelas com reflexo no banco de dados, limitando as consultas ao banco a apenas os registros da página atual.  
REQNAOFUNC 10. : Utilizou uma anotação específica com regra própria para validar a matrícula.

### Recomendações

- Utilizar um nome próprio para o sistema. Evite: “projetospring”, “projeto fred”, “projeto pweb2”, “projeto”. Prefira: “Saturno”, “eColegiado”, etc.
- Usar Maven e Git.
- Uma apresentação deve ser realizada pela equipe demonstrando todas as funcionalidades (ou bugs) do sistema. Funcionalidade não apresentada será considerada não implementada. A apresentação deve mostrar também a arquitetura do código fonte. Não precisa mostrar classe a classe, só o esquema geral (“aqui implementamos controladores para não pôr a lógica dentro dos controladores Spring”, “não vamos mostrar os Repositories porque o Spring gera automático” etc).
- Código sem apresentar funcionalidades com telas, registro no banco etc. não será considerado apto para a nota. É preciso mostrar as funcionalidades plenas no sistema e não classes Java que não entregam nenhum valor ao usuário final (eu).
- O repositório Git deve conter commits de TODOS os membros da equipe como registro da participação na implementação do código.
- Não comecem a fazer a 5 dias do prazo**, pois simplesmente não dará tempo de entregar. O melhor momento para começar foi ontem, o segundo melhor é agora.
- Não será possível dilatar muito o prazo da reposição pois a entrega ocorrerá na última semana de aula do semestre letivo.

### Pontuação para correção do projeto nas duas etapas

Itens da Etapa I	Pontos
1) Implementou o REQFUNC 1	10
2) Implementou o REQFUNC 2	10
3) Implementou o REQFUNC 3	15
4) Implementou o REQFUNC 7	15
5) Implementou o REQFUNC 8	10
6) Implementou o REQFUNC 13	05
7) Implementou o REQFUNC 14	15
8) Implementou o REQFUNC 15	05
9) Cumpriu o REQNAOFUNC 2	05
10) Cumpriu o REQNAOFUNC 5	05
11) Cumpriu o REQNAOFUNC 6	05
<b>Total de Pontos</b>	<b>100</b>



Itens da Etapa II	Pontos
1) Implementou o REQFUNC 4	10
2) Implementou o REQFUNC 5	05
3) Implementou o REQFUNC 6	10
4) Implementou o REQFUNC 9	15
5) Implementou o REQFUNC 10	05
6) Implementou o REQFUNC 11	15
7) Implementou o REQFUNC 12	05
8) Implementou o REQFUNC16	05
9) Cumpriu o REQNAOFUNC 7	10
10) Cumpriu o REQNAOFUNC 8	10
11) Cumpriu o REQNAOFUNC 9	05
12) Cumpriu o REQNAOFUNC10	05
<b>Total de Pontos</b>	<b>100</b>



II Avaliação de Aprendizagem – Entrega da Etapa I

Equipe							
REQFUNC 1 10 Aluno cad proc							
REQFUNC 1 10 Aluno lista proc							
REQFUNC 1 15 Prof lista proc atribuidos							
REQFUNC 1 15 Coord lista proc							
REQFUNC 1 10 Coord dist proc							
REQFUNC 13 05 Admin crud coleg							
REQFUNC 14 15 Admin crud pessoas							
REQFUNC 15 05 Coord crud assunto							
REQFUNC 12 05 CSS							
REQFUNC 15 05 Validadores							
REQFUNC 1 05 P=R=G							

II Avaliação de Aprendizagem – Entrega da Etapa II



Equipe						
REQFUNC 4 Prof consulta reuniões (filtro) 10						
REQFUNC 5 Prof vota com texto 10						
REQFUNC 9 Coord cria sessão do colegiado e def pauta 15						
REQFUNC 10 Coord inicia sessão 10						
REQFUNC 11 Coord apregoa processo em sessão 15						
REQFUNC 12 Coord finaliza sessão 05						
REQNAOFUNC 7 Layouts e Frags 10						
REQNAOFUNC 8 Auth 10						
REQNAOFUNC 9 Paginação 05						
REQNAOFUNC 10 Validador matrícula 10						
Total						