## Análise Crítica dos Requisitos do Sistema de Avaliação Docente

**Problemas e Fragilidades nos Requisitos Atuais**

**Revisão Geral:** Os requisitos listados fornecem uma base para o sistema, mas apresentam algumas lacunas e suposições que precisam ser esclarecidas. A seguir, analisamos cada requisito identificando possíveis falhas ou pontos frágeis, bem como considerações adicionais:

* **RF01 – Login de usuários:** O login por e-mail e senha para alunos, gestores e diretores é adequado, porém notamos que **docentes (professores)** e **administradores** não foram mencionados. Supondo que docentes não tenham acesso ao sistema (apenas são avaliados) e que o administrador seja um usuário técnico, isso deve ser explicitado. Além disso, é importante ressaltar que **senhas não devem ser armazenadas em texto puro**; a melhor prática é armazená-las usando **hash seguro**, não criptografia reversível ([Password Storage - OWASP Cheat Sheet Series](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html" \l ":~:text=Hashing%20and%20encryption%20can%20keep,should%20be%20hashed%2C%20NOT%20encrypted)). Isso garante que mesmo em caso de vazamento do banco de dados, as senhas não sejam obtidas diretamente. Portanto, o RF01 deve incluir a regra de armazenamento seguro de senha (ver RNF03).
* **RF02 – Aluno visualiza lista de docentes vinculados ao curso:** Pressupõe-se que cada aluno pertença a um *curso* (por exemplo, seu curso de graduação) e que o sistema mostre todos os docentes associados àquele curso. Uma **fragilidade** aqui é *como determinar os docentes vinculados*: seria melhor listar apenas os docentes com os quais o aluno teve aulas naquele período, para avaliações mais relevantes. Do jeito que está, um aluno poderia ver (e ter que avaliar) todos os professores do curso, mesmo aqueles que não lhe ministraram aula – isso pode distorcer os resultados. Seria necessário ou **integrar dados acadêmicos** (matrículas em turmas) ou pelo menos assumir que todos os alunos de um curso veem todos os professores do curso. Essa suposição deve ser clarificada. Além disso, deve-se considerar *como o vínculo docente-curso será cadastrado* (ver RF10) e se o aluno pode filtrar ou buscar um docente específico na lista (usabilidade).
* **RF03 – Aluno responde formulário obrigatório de múltipla escolha:** Este requisito indica que haverá um **formulário de avaliação padronizado**, com perguntas de múltipla escolha (por exemplo, escalas de 1 a 5). Pontos a considerar:
  + Quantas perguntas? Todas são obrigatórias? Pelo texto, sim (“formulário obrigatório” implica que o aluno deve responder a todas as questões fechadas).
  + Deveria ser mencionado que as questões são definidas pelo administrador (relacionando com RF09).
  + Também é importante esclarecer se o aluno responde **um formulário por docente**. Presume-se que sim (cada professor avaliado individualmente). Logo, o fluxo seria: Aluno seleciona um docente da lista (RF02) e responde ao formulário referente àquele docente.
* **RF04 – Aluno deixa comentário opcional por avaliação:** Aqui identifica-se que, além das questões objetivas, o aluno pode fornecer um **comentário textual opcional** sobre o docente. Isso é positivo (combina avaliação quantitativa e qualitativa). É importante assegurar que esses comentários **permaneçam anônimos** quando exibidos a gestores/diretores, para que alunos se sintam seguros em se expressar. O requisito não menciona anonimato, o que é **uma omissão séria**: em avaliações docentes reais, costuma-se garantir que as respostas individuais sejam anônimas (especialmente os comentários, para não identificar o aluno). Portanto, é recomendável acrescentar um requisito de **anonimato** das avaliações, garantindo que os professores/gestores não saibam qual aluno fez qual comentário. Estudos de sistemas de avaliação enfatizam a necessidade de manter o anonimato dos alunos, ao mesmo tempo em que se acompanha quem já respondeu para fins de taxa de participação ([Developing Online Course and Lecturer Evaluation System using DBase](https://utpedia.utp.edu.my/3933/1/13995_IZYAN_IZZATI_BAHARUDDIN_dissertation_5562.pdf" \l ":~:text=universities,a%20mechanism%20for%20report%20generation)).
* **RF05 – Impedir avaliação duplicada por aluno/docente/período:** Esse requisito visa prevenir que um mesmo aluno avalie o mesmo professor mais de uma vez no mesmo **período**. Pressupõe a definição clara de “período” – provavelmente um semestre letivo ou ano. Deve-se clarificar o que configura o período (por exemplo: 2025.1, 2025.2, etc.). Tecnicamente, isso implica uma **restrição de unicidade** no banco de dados (chave composta de aluno+docente+período) ou validação na aplicação. Um ponto a corrigir: se um aluno submeter uma avaliação e quiser alterá-la logo em seguida (por ter cometido um erro, por exemplo), o sistema permitiria edição? Pelo RF05, não – ele bloqueia duplicatas. Seria útil permitir **atualização da avaliação** dentro do prazo, em vez de bloqueio absoluto, ou pelo menos comunicar claramente que uma vez enviada, não pode refazer. Além disso, caso um professor lecione em dois cursos diferentes, um aluno de curso A não deve conseguir avaliá-lo no contexto do curso B; o sistema deve garantir que apenas avalie professores vinculados ao seu curso (relacionado ao RF02).
* **RF06 – Gestor acessa médias de avaliações dos docentes do seu curso:** Aqui identificamos que o *Gestor* (provavelmente coordenador ou chefe de departamento) pode ver os **resultados agregados** das avaliações para os docentes daquele curso. “Médias” sugere que o sistema calculará a média das notas das questões de múltipla escolha para cada docente. Pontos de atenção:
  + Que tipo de média? Provavelmente a média das pontuações de cada pergunta ou um agregado geral. Seria útil especificar se o gestor vê, por exemplo, a média de cada questão por professor ou uma nota global por professor.
  + **Análise de fragilidade:** se um professor teve poucos alunos, mostrar a média pode expor quem respondeu. Idealmente, deveria haver um número mínimo de respostas para exibir resultados, preservando anonimato (por exemplo, se apenas 1 aluno avaliou um professor, o gestor veria a média daquela única avaliação – isso quebra a anonimidade). Esse tipo de regra poderia ser considerada.
  + Supõe-se que o gestor só veja os dados do seu curso. Então deve haver um controle de acesso para que um gestor de Curso X não veja docentes de Curso Y. Isso depende de como o vínculo gestor-curso for implementado (ver RF10 e considerações de modelo).
* **RF07 – Diretor acessa médias de todos os docentes:** O Diretor (possivelmente diretor da instituição ou unidade) teria acesso a resultados agregados de *todos os docentes*, independentemente do curso. Isso é lógico se o diretor tem supervisão geral. Deve-se garantir que o diretor veja apenas agregados anônimos. Uma fragilidade potencial: se o diretor for responsável por múltiplos cursos ou departamentos, o volume de dados pode ser grande – talvez filtros ou seleção por curso seriam úteis (não citado, mas poderia melhorar a usabilidade). Contudo, como requisito mínimo, está ok. Novamente, deveria seguir as mesmas regras de anonimato e apresentação que para gestores.
* **RF08 – Dashboard visual para gestores e diretores:** Este requisito indica a necessidade de uma **visualização gráfica** (dashboard) dos resultados, provavelmente incluindo gráficos, indicadores de satisfação média, comparativos, etc. Pontos a considerar:
  + É um requisito bastante amplo (“dashboard visual” pode abranger muitas coisas). Em um **MVP (Produto Mínimo Viável)** talvez bastasse tabelas simples ou gráficos básicos. Deve-se gerenciar expectativas aqui – se for MVP, o dashboard inicialmente poderia ser simples (por exemplo, um gráfico de barras por pergunta ou por professor).
  + Seria útil especificar *que tipo de visualizações* são esperadas (média por professor, distribuição de notas, evolução temporal se houver dados históricos, etc.), embora isso possa ficar para a fase de design detalhado.
  + Do ponto de vista de implementação, gráficos podem ser gerados via biblioteca web; isso implica um requisito não-funcional de compatibilidade de front-end, mas nada problemático.
* **RF09 – Cadastro de perguntas pelo administrador:** Prevê que um **Administrador do sistema** possa gerenciar as perguntas do formulário. Ou seja, o questionário não é “fixo no código”, mas dinâmico. Essa é uma funcionalidade essencial para permitir ajustes no instrumento de avaliação sem precisar alterar o sistema. Fragilidades:
  + Deve-se definir se o administrador pode **criar, editar e remover perguntas** (CRUD completo) ou apenas inserir novas. Provavelmente CRUD.
  + Alterações nas perguntas ao longo do tempo podem impactar comparabilidade de médias históricas – mas, como o escopo não menciona histórico, talvez não seja relevante agora.
  + Também não foi mencionado, mas assume-se que **todas as avaliações usam o mesmo conjunto de perguntas** (um formulário padrão). Se no futuro quiserem formulários diferentes por curso ou departamento, o modelo precisaria suportar isso – no MVP, parece não ser o caso.
  + Deve-se garantir que perguntas ativas sejam refletidas no formulário que o aluno responde e associadas às respostas (ver modelo de dados proposto).
* **RF10 – Associação docente-curso:** Este requisito sugere que o sistema deve manter o vínculo entre cada professor (docente) e o curso em que leciona. Provavelmente será usado para:
  + Filtrar quais docentes aparecem para cada aluno (RF02).
  + Determinar quais docentes cada gestor pode ver (RF06).
  + A formulação “associação” indica que um docente pode estar associado a um ou **vários cursos** (por exemplo, um professor pode lecionar em dois cursos/programas diferentes). Assim, o relacionamento docente-curso no modelo deve ser **N:N** (muitos-para-muitos) a priori.
  + Deve ficar claro **quem realiza essa associação**. É presumível que seja o administrador, já que faz parte do gerenciamento de cadastros. Logo, **falta um requisito de cadastro de docentes e cursos** em si: o RF10 fala em associá-los, mas não menciona como os docentes e cursos entram no sistema. Provavelmente é necessário um RF adicional para **cadastrar (CRUD) os docentes e cursos** no sistema, tarefa também do administrador. Sem isso, o sistema não teria base de dados inicial de professores ou cursos.
  + Outra consideração: se a instituição já possui esses dados em outro sistema, talvez a ideia fosse importá-los. Mas como não foi dito, devemos assumir que o próprio sistema terá telas para cadastrar cursos e professores manualmente (ou pelo menos importar via planilha).
* **RNF01 – Acesso via navegador:** Indica que o sistema será uma aplicação **Web** (intranet possivelmente, já que é local/interno). Isso é adequado. Só vale ressaltar compatibilidade com navegadores comuns e diferentes dispositivos (relacionado ao RNF05 de responsividade). Não há falhas aqui, é um requisito padrão.
* **RNF02 – Desenvolvimento em formato de MVP:** Isso destaca que o projeto será implementado inicialmente como um **MVP (Minimum Viable Product)**, ou seja, priorizando o essencial para uso rápido e depois evoluindo. Essa abordagem é positiva, mas deve-se tomar cuidado para não deixar de fora nenhum dos requisitos funcionais mínimos indispensáveis. A lista atual de RFs já é relativamente completa para um sistema de avaliação básico. Talvez a ideia seja implementar primeiro as funções básicas (login, avaliação, cálculo de média) e posteriormente expandir (por exemplo, dashboards mais elaborados podem ficar para depois da versão inicial). Nenhuma correção necessária, mas lembrar que MVP implica possivelmente *não* implementar todas as sofisticações de primeira (por exemplo, se o prazo estiver apertado, o “dashboard visual” poderia ser simplificado inicialmente).
* **RNF03 – Autenticação segura com criptografia de senha:** Conforme mencionado na análise do RF01, autenticação segura é obrigatória. Aqui o termo *criptografia de senha* deve ser interpretado com cuidado – o ideal é **hash irreversível com sal** em vez de criptografia bidirecional ([Password Storage - OWASP Cheat Sheet Series](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html" \l ":~:text=Hashing%20and%20encryption%20can%20keep,should%20be%20hashed%2C%20NOT%20encrypted)). Então, uma melhoria de redação seria: “armazenamento seguro das senhas (hash seguro, p.ex. bcrypt ou Argon2, em vez de senha em texto claro)”. Fora isso, esse requisito reforça a importância de segurança, sem falhas conceituais.
* **RNF04 – Banco de dados relacional:** Adequado, já que o sistema lida com dados estruturados (usuários, cursos, avaliações, perguntas, etc.) e consultas (médias). Um SGBD relacional (MySQL, PostgreSQL, etc.) é apropriado. Talvez mencionar **qual SGBD** ou se há preferência, mas não é obrigatório no requisito. Também pode-se considerar requisitos de integridade referencial (que um SGBD relacional naturalmente suporta).
* **RNF05 – Interface responsiva e acessível:** Muito pertinente – significa que o front-end deve funcionar em diferentes tamanhos de tela (desktop, mobile) e seguir boas práticas de acessibilidade (por exemplo, suporte a leitores de tela, alto contraste, etc.). Não há falha aqui; apenas nota-se que é um requisito amplo e qualitativo, cuja implementação deve acompanhar guias de UX. Em um MVP, garantir responsividade básica (via CSS frameworks, por exemplo) é factível.
* **RNF06 – Tempo de resposta inferior a 2 segundos:** Esse requisito de desempenho busca garantir uma boa experiência ao usuário. Dois segundos é razoável para páginas carregarem (de preferência bem menos na maioria das ações). Em um sistema local com poucos usuários simultâneos, não deve ser um problema. Entretanto, é bom especificar em que contexto: 2s para carregar qualquer página ou relatório dentro de condições normais de carga. É um requisito não-funcional típico. Deve-se cuidar de otimizar consultas (especialmente cálculo de médias – embora sejam simples – e carregamento do dashboard). Provavelmente está tudo bem aqui, apenas ciente de monitorar se gráficos pesados ou consultas com muitos dados no futuro mantêm esse tempo. Neste MVP, o volume de dados será baixo inicialmente.

**Resumo das Fragilidades:** Em suma, os pontos que requerem correção ou complementação são: **(1)** Inclusão de requisitos para **anonimato** das avaliações e proteção da identidade dos alunos; **(2)** Garantir funcionalidades de **gestão de dados básicos** (cursos, docentes, usuários) pelo administrador que não foram explicitadas; **(3)** Especificar melhor a relação entre aluno, professor e curso no ato da avaliação (evitar que aluno avalie professor de outro curso, por exemplo); **(4)** Considerar se professores poderão ver seus resultados (o sistema atual não menciona isso – possivelmente uma escolha deliberada, mas vale discutir); **(5)** Ajustes de segurança e implementação conforme boas práticas (senhas com hash, etc., que complementam RNF03); **(6)** Clarificar detalhes de agregação de resultados (médias) e critérios de exibição para evitar dedução de respondentes únicos.

**Contrapontos e Perspectivas Alternativas**

Antes de propor a versão aprimorada dos requisitos, é importante discutir algumas **perspectivas alternativas** e decisões de design não explícitas nos requisitos originais:

* **Acesso dos Docentes aos Resultados:** Os requisitos originais omitem qualquer interface para que o professor (docente) avaliado veja suas próprias avaliações. Em muitas instituições, é prática que os docentes recebam feedback de suas avaliações – geralmente um relatório consolidado entregue pela coordenação ou disponível em sistema. Poderia-se cogitar um requisito permitindo que o **docente faça login e veja suas médias e comentários** (após o período de avaliações encerrar). No entanto, isso pode ser sensível: algumas instituições preferem que apenas gestores tenham acesso direto, repassando aos docentes de forma moderada. **Contraponto:** Incluir os docentes como usuários do sistema com acesso aos seus resultados aumentaria transparência e agilidade, mas exigiria adicionar todo um papel no controle de acesso e possivelmente filtrar comentários ofensivos antes. Para o MVP, talvez mantenha-se a decisão de **não dar acesso direto aos docentes**, deixando a visualização apenas para gestores/diretores, com a ideia de que estes irão compartilhar os feedbacks com os professores conforme a política interna.
* **Anonimato e Confidencialidade:** Reforçando o ponto crítico: o sistema deve **assegurar anonimato**. Isso não estava explícito, mas é fundamental. Os formulários impressos tradicionais garantem isso, e numa aplicação web deve ser igual. Uma alternativa a se considerar é implementar o sistema de forma que, embora se rastreie quem respondeu (para evitar duplicatas), o **relatório apresentado não identifique alunos individualmente**. Por exemplo, os comentários poderiam ser exibidos *sem nenhuma informação pessoal*. Poderia-se até embaralhar a ordem de apresentação ou agregar respostas para evitar qualquer associação. Essa é uma garantia ética importante e deveria virar um requisito não-funcional (privacidade).
* **Rastrear participação e incentivar respostas:** O sucesso de um sistema de avaliação docente depende da adesão dos alunos. Não está nos requisitos, mas como *perspectiva futura*, poderia-se considerar: envio de lembretes automáticos para quem não preencheu, ou bloqueio de algo (por exemplo, algumas instituições bloqueiam a visualização de notas finais do aluno até ele preencher as avaliações, para garantir participação ([Developing Online Course and Lecturer Evaluation System using DBase](https://utpedia.utp.edu.my/3933/1/13995_IZYAN_IZZATI_BAHARUDDIN_dissertation_5562.pdf" \l ":~:text=the%20evaluations,have%20more%20time%20to%20really)) ([Developing Online Course and Lecturer Evaluation System using DBase](https://utpedia.utp.edu.my/3933/1/13995_IZYAN_IZZATI_BAHARUDDIN_dissertation_5562.pdf" \l ":~:text=restricted%20students%20from%20viewing%20the,have%20more%20time%20to%20really))). Como isso não foi mencionado e pode ser polêmico, talvez fique fora do escopo inicial. Mas vale o gestor saber quem já respondeu ou não? Se sim, precisaria de um relatório de participação (por aluno). No MVP, possivelmente não faremos isso, mas é uma ideia a discutir com os stakeholders.
* **Integração com outros sistemas:** Não citado, mas em contexto real, dados de alunos, cursos, docentes poderiam vir de um sistema acadêmico existente. No MVP, assumimos tudo será gerenciado internamente (cadastro manual). Entretanto, uma visão alternativa é integrar via importação ou sincronização de dados para evitar retrabalho. Novamente, isso adiciona complexidade e talvez fuja do escopo inicial, mas fica como sugestão futura.
* **Método de Cálculo das Métricas:** Ao mencionar "médias", assume-se média aritmética simples. Alternativamente, poderia-se calcular distribuição de respostas, medianas, etc. No MVP, a média atende, mas deve-se ter cuidado: médias podem mascarar variabilidade. Um dashboard mais avançado poderia mostrar a **distribuição das notas** por pergunta (ex.: quantos marcaram 5, 4, 3, etc.), para contexto. Uma melhoria incremental seria incluir essa visão no futuro (talvez um gráfico de barras para cada questão no dashboard, ao invés de somente a média numérica).
* **Escala de Avaliação:** “Múltipla escolha” sugere uma escala (por exemplo 1 a 5). Deve-se decidir isso. Uma escala Likert de 5 pontos é comum. É bom que o sistema permita ao admin cadastrar as perguntas **com opções fixas** (talvez implicitamente sempre 1-5). Alternativamente, poder-se-ia querer perguntas do tipo “Sim/Não” ou “A, B, C, D, E”. Para simplicidade, todas as perguntas podem usar a mesma escala padrão (e.g., 1 a 5 onde 5 = ótimo). Essa decisão impacta implementação mas não foi explicitada. Provavelmente ficará definida no conteúdo das perguntas (ex.: “Avalie de 1 a 5…”).
* **Unicidade de Gestor por Curso:** Supõe-se que cada curso tenha um único gestor responsável. Os requisitos não contemplam múltiplos gestores por curso (por exemplo, dois coordenadores compartilhando a gestão). Em casos reais, pode haver vice-coordenadores. No MVP, assumiremos **um curso = um gestor**. Se essa suposição não valer, o modelo de dados precisa suportar gestor-curso N:N. Novamente, para simplificar, consideramos 1:1, mas deixar em aberto para alteração futura se necessário.
* **Segurança e Permissões:** Além da criptografia de senha (RNF03), vale pensar em **controle de acesso** no front-end/back-end – garantir que um aluno não acesse páginas de gestor via URL, etc. Isso é implementação, mas crucial. Talvez adicionar um RNF de *controle de autorização por perfil*, embora isso seja esperado implicitamente.
* **Registro de Atividades:** Não solicitado, mas poderia ser útil registrar data/hora de cada avaliação submetida, para eventualmente saber prazos ou auditar duplicidades. No modelo de dados, poderíamos ter um timestamp na avaliação. Não é obrigatório, mas fácil de implementar e pode ajudar em análise (ex.: período de respostas, ou para confirmar que o período correto foi usado).

Considerando todos esses contrapontos, iremos **corrigir e melhorar** os requisitos na próxima seção, incorporando especialmente: anonimato, cadastro de docentes/cursos, e ajustes de redação para maior clareza.

**Versão Aprimorada dos Requisitos**

Abaixo apresentamos uma versão revisada e aprimorada dos requisitos, incorporando as correções e esclarecimentos discutidos:

**Requisitos Funcionais (Aprimorados):**

1. **RF01 – Autenticação de Usuários:** O sistema deve permitir o login de usuários autorizados (alunos, gestores, diretores e administrador) mediante e-mail e senha. As senhas devem ser tratadas de forma segura (armazenadas como hash criptográfico). Cada tipo de usuário terá acesso apenas às funções correspondentes ao seu papel (perfil).
2. **RF02 – Listagem de Docentes para Avaliação:** O aluno deve visualizar a lista de docentes associados ao seu curso (ou às suas turmas, no período vigente) que ele deve avaliar. Essa lista deve ser filtrada de forma que o aluno veja **apenas docentes relevantes** (por exemplo, que lecionaram para sua turma naquele semestre).
3. **RF03 – Realização de Avaliação (Questionário):** O aluno pode selecionar um docente da lista e responder um **formulário de avaliação** composto por perguntas de múltipla escolha obrigatórias (por exemplo, escala de 1 a 5) definidas pelo administrador. Todas as questões fechadas devem ser respondidas para que a avaliação seja válida.
4. **RF04 – Comentário Opcional:** Ao finalizar a avaliação de um docente, o aluno pode, **opcionalmente**, escrever um comentário textual livre sobre o desempenho daquele docente. O sistema deve permitir enviar a avaliação mesmo sem comentário.
5. **RF05 – Controle de Unicidade de Avaliação:** O sistema deve impedir que um mesmo aluno envie mais de uma avaliação para o mesmo docente no mesmo período letivo. Ou seja, **cada combinação aluno-docente-período só pode gerar uma avaliação**. (Se necessário, pode-se permitir que o aluno edite sua avaliação enquanto o período de avaliação estiver aberto, em vez de submeter múltiplas.)
6. **RF06 – Anonimato das Avaliações:** As respostas individuais dos alunos (incluindo comentários) **devem permanecer anônimas** para os avaliados e para quem visualizar os resultados. Gestores e diretores não devem conseguir identificar quais alunos deram quais notas ou comentários. O sistema só usará a identificação do aluno para fins de controle (RF05 e relatórios de participação), mas nunca exibirá essa associação nos relatórios de resultados.
7. **RF07 – Consolidação de Resultados por Docente:** O sistema deve calcular e armazenar os resultados agregados de cada docente avaliado em cada período – principalmente as **médias** das notas de cada pergunta (ou uma média geral) e compilar os comentários recebidos.
8. **RF08 – Acesso do Gestor aos Resultados do Curso:** O gestor (coordenador) deve conseguir acessar os resultados consolidados das avaliações dos docentes do **seu curso**. Isso inclui visualizar as médias das notas de cada docente em seu curso (idealmente por pergunta) e os comentários recebidos para cada docente sob sua coordenação.
9. **RF09 – Acesso do Diretor aos Resultados Gerais:** O diretor deve conseguir visualizar os resultados consolidados de **todos os docentes da instituição** (todos os cursos). O diretor pode acessar as médias e feedbacks de qualquer docente. Se houver muitos dados, o sistema deve permitir filtrar ou selecionar por curso/departamento para facilitar a visualização (opcional no MVP).
10. **RF10 – Dashboard Gerencial:** O sistema deve fornecer um **dashboard visual** para gestores e diretores, apresentando os resultados das avaliações de forma gráfica e intuitiva. Por exemplo, gráficos que mostrem a média de cada docente, comparação entre turmas ou cursos, distribuição de respostas por pergunta, etc. *(No MVP, pode ser uma página de relatório simples com alguns gráficos básicos.)*
11. **RF11 – Cadastro e Gerenciamento de Perguntas:** O administrador do sistema deve poder **cadastrar, editar, remover** perguntas no banco de questões utilizadas no formulário de avaliação. As perguntas são de múltipla escolha e, ao serem criadas, terão um conjunto padrão de opções de resposta (por exemplo, escala de 1 a 5, onde 1 = muito insatisfeito e 5 = muito satisfeito). Qualquer alteração no conjunto de perguntas deve refletir para as novas avaliações (sem afetar dados passados já coletados).
12. **RF12 – Cadastro de Cursos e Docentes:** O administrador deve poder cadastrar novos **cursos** (ex.: “Engenharia de Produção”) e novos **docentes** (professores) no sistema, informando seus dados básicos (nome, etc.).
13. **RF13 – Associação Docente–Curso:** O administrador deve poder associar cada docente a um ou mais cursos nos quais ele leciona. Essa associação é necessária para que alunos vejam os docentes corretos (RF02) e para que gestores tenham acesso apenas aos docentes do seu curso (RF08). Deve ser possível adicionar ou remover cursos vinculados a um docente a qualquer momento. *(Se um docente lecionar em dois cursos, ele será avaliado separadamente em cada um — o sistema pode, neste caso, tratá-lo como duas entradas de avaliação distintas por curso.)*
14. **RF14 – Gestão de Usuários e Permissões:** (Opcional no MVP) O administrador pode gerenciar contas de usuário do sistema, definindo quem são os gestores, diretores e eventualmente vinculando gestores aos seus cursos. Isso inclui criar contas para novos usuários desses perfis e garantir que cada gestor esteja associado ao curso correto. *(Observação: Alternativamente, pode-se pré-cadastrar no banco os logins de gestores/diretor manualmente.)*

**Requisitos Não Funcionais (Aprimorados):**

1. **RNF01 – Aplicação Web (Acesso via Navegador):** O sistema deve ser disponibilizado como uma aplicação web, acessível através dos navegadores mais comuns (Chrome, Firefox, etc.), preferencialmente compatível com as últimas versões. Não haverá necessidade de instalação de software cliente – apenas conexão à rede local da instituição.
2. **RNF02 – Arquitetura MVP Evolutiva:** O desenvolvimento seguirá o formato de **Produto Mínimo Viável (MVP)**, entregando inicialmente um conjunto básico de funcionalidades plenamente operacional e, posteriormente, incorporando melhorias (por exemplo, aperfeiçoamentos no dashboard, novas funções como notificações de e-mail, etc.) de forma incremental. Isso implica um design flexível que permita ajustes rápidos conforme feedback dos usuários.
3. **RNF03 – Segurança da Autenticação:** As senhas dos usuários devem ser armazenadas de forma **segura** (hash com algoritmo robusto e sal único por senha). A autenticação deve ocorrer sobre conexão segura (ex.: HTTPS se houver rede aberta) para proteger as credenciais em trânsito. Além disso, deve haver proteção contra tentativas de login inválidas (por exemplo, bloquear após X tentativas falhas para evitar brute force). ([Password Storage - OWASP Cheat Sheet Series](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html" \l ":~:text=Hashing%20and%20encryption%20can%20keep,should%20be%20hashed%2C%20NOT%20encrypted))
4. **RNF04 – Banco de Dados Relacional:** O sistema utilizará um SGBD relacional para persistência dos dados, garantindo integridade referencial entre entidades (por exemplo, não permitir avaliação referenciar um aluno inexistente, etc.). Todos os dados de avaliações (respostas) e cadastros devem ser armazenados neste banco. Deve-se considerar backups periódicos e recuperação de dados (confiabilidade), embora isso possa ser procedimental.
5. **RNF05 – Interface Responsiva e Acessível:** A interface do usuário deverá ser amigável, responsiva a diferentes tamanhos de tela (desktop, tablets, celulares) e seguir boas práticas de UX acessível (textos com contraste adequado, suporte a navegação por teclado, etc.). Isso visa permitir que alunos respondam às avaliações em diversos dispositivos e que nenhum usuário seja impedido de usar o sistema por deficiência visual/motora.
6. **RNF06 – Desempenho e Escalabilidade:** O sistema deve apresentar tempo de resposta de, no máximo, 2 segundos para as principais operações (carregar a lista de docentes, submeter um formulário de avaliação, carregar o dashboard, etc.) sob carga normal. Em caso de muitos usuários simultâneos (por ex., vários alunos avaliando ao mesmo tempo em um laboratório), o sistema ainda deve se manter responsivo, talvez requerendo otimizações de consulta (uso de índices para cálculo de médias, paginação de resultados no dashboard, etc.).
7. **RNF07 – Auditoria e Logs (Opcional):** *(Requisito adicional sugerido)* O sistema deve manter logs básicos de atividades (logins, submissões de formulário, ações administrativas) para possibilitar auditoria e diagnóstico de problemas. Esses logs não conterão dados privados dos alunos, apenas registros técnicos e timestamps das operações principais.

*(Os requisitos acima refletem melhorias de clareza e completude em relação à lista original, adicionando pontos como anonimato, cadastro de dados básicos e outros aspectos de segurança e usabilidade discutidos.)*

**Diagrama de Casos de Uso do Sistema**

*Diagrama de Casos de Uso proposto para o sistema de avaliação docente.* Neste diagrama, representamos os **atores** do sistema – Aluno, Gestor, Diretor e Administrador – e os **casos de uso** (funcionalidades) correspondentes. Observe que o caso de uso "**Preencher avaliação (questionário)**" engloba responder às perguntas obrigatórias; o caso de uso "**Adicionar comentário**" é uma extensão opcional daquele (indicado pela relação <<extend>> no diagrama). Cada ator só pode realizar os casos de uso relacionados ao seu perfil. Por exemplo, o **Aluno** pode *Visualizar docentes do curso* e *Preencher avaliação*, enquanto o **Gestor** pode *Consultar médias do curso* e acessar o *Dashboard*. O caso de uso de **Login** é comum a todos os atores (todos precisam se autenticar). O **Administrador** possui casos de uso especiais de manutenção: *Gerenciar perguntas* (cadastro/edição de perguntas) e *Associar docente a curso* (vincular professores aos cursos).

No diagrama, as linhas sólidas conectando atores a casos de uso representam as interações primárias. O ator Diretor tem permissões similares ao Gestor no que tange a consultar resultados, mas em escopo mais amplo (todos os docentes). Note que não inserimos um ator "Docente" no diagrama, pois conforme os requisitos aprimorados, o professor avaliado em si não interage diretamente com o sistema (embora isso possa ser revisto futuramente). O foco está nas funcionalidades solicitadas: desde o login, passando pela avaliação feita pelo aluno, até a visualização de relatórios pelos gestores/diretor e a administração do questionário pelo administrador.

**Modelo Entidade-Relacionamento (ER) do Sistema**

*Modelo ER proposto mostrando as principais entidades e relacionamentos.* As entidades principais identificadas são: **Aluno**, **Gestor**, **Diretor**, **Administrador**, **Docente**, **Curso**, **Avaliação**, **Pergunta** e **Resposta**. Cada uma corresponde a dados que o sistema precisará armazenar:

* Atores do sistema com login (Aluno, Gestor, Diretor, Administrador) foram representados separadamente aqui para clareza, cada um com atributos de identificação (*ID, Nome, Email, Senha*). Em uma implementação real, poderíamos unificá-los em uma única tabela de usuários com um atributo de tipo/perfil, mas conceitualmente estão distinguidos. Observa-se que tanto **Aluno** quanto **Gestor** têm um atributo de **CursoID (FK)** indicando a que curso estão vinculados (aluno estuda em um curso; gestor coordena um curso). Assim, um curso tem muitos alunos e (presumivelmente) um gestor – no diagrama, o relacionamento Curso–Aluno é 1:N, e Curso–Gestor foi modelado como 1:1 (um gestor por curso; um gestor gerencia um curso).
* A entidade **Docente** representa os professores avaliados. Eles têm seus dados (ID, Nome, Email). A relação entre **Docente** e **Curso** é *muitos-para-muitos* (N:N), já que um docente pode lecionar em vários cursos e cada curso tem vários docentes. No modelo, isso aparece como dois *relacionamentos* distintos: Docente -> Curso (N:N). Implementacionalmente, seria uma tabela associativa Docente\_Curso contendo as chaves dos dois, mas conceitualmente indicamos com a notação de cardinalidade (N próximo a Docente e N próximo a Curso).
* A entidade **Pergunta** guarda as perguntas do formulário de avaliação (cada pergunta com um ID e texto). A entidade **Avaliação** representa uma avaliação feita por um aluno sobre um docente em um período. Ela tem um ID, o atributo *Período* (p.ex. “2025.1”), o campo *Comentário*, e chaves estrangeiras para o **Aluno** que fez a avaliação e o **Docente** avaliado. Há uma relação *Aluno 1:N Avaliação* (um aluno pode fazer várias avaliações – uma para cada professor que teve naquele período) e *Docente 1:N Avaliação* (um docente recebe várias avaliações – uma de cada aluno que o avaliar). Assim, a entidade Avaliação é central ligando alunos e docentes. A restrição de **unicidade aluno-docente-período** citada em RF05 seria garantida impondo que não haja duas avaliações com mesmos AlunoID, DocenteID e Período. Poderíamos compor a chave primária por esses três campos, mas aqui optamos por um ID para simplificar e consideramos a unicidade como restrição lógica.
* A entidade **Resposta** detalha as respostas às perguntas de cada avaliação. Como cada avaliação consiste em responder várias perguntas, modelamos **Resposta** com uma chave composta (AvaliaçãoID + PerguntaID) e um campo *Valor* (que armazenará a escolha do aluno naquela pergunta – por exemplo, um número de 1 a 5). A relação **Avaliação 1:N Resposta** indica que uma Avaliação terá várias Respostas (uma por pergunta), e **Pergunta 1:N Resposta** indica que uma Pergunta aparece em várias respostas (ao longo de muitas avaliações). Na prática, a tabela Resposta armazena o resultado de cada pergunta de cada avaliação individual. Os comentários opcionais, como fazem parte da avaliação como um todo, ficam no campo *Comentário* de Avaliação, não na tabela Resposta.

Algumas observações sobre o modelo: As entidades **Gestor**, **Diretor** e **Administrador** não possuem relações diretas com outras entidades (além de Gestor com Curso) porque servem principalmente para acesso ao sistema e definição de permissões. Por simplicidade, não representamos uma entidade unificada de “Usuário” com herança, mas isso poderia ser considerado em um design físico. O importante é que exista uma forma de distinguir e vincular os papéis: no modelo apresentado, o **Curso** tem um Gestor (via atributo CursoID em Gestor ou poderia ser inverso via GestorID em Curso), e sabemos que o Diretor e Administrador não precisam de vínculos (Diretor tem acesso global, Admin apenas gerencia configurações).

**Integridade referencial:** No banco de dados relacional, teremos chaves estrangeiras implementando os relacionamentos acima. Por exemplo, Avaliação.AlunoID referencia Aluno.AlunoID; Avaliação.DocenteID referencia Docente.DocenteID; Gestor.CursoID referencia Curso.CursoID; Resposta.AvaliacaoID e Resposta.PerguntaID referenciam Avaliação e Pergunta, respectivamente. Assim garantimos consistência (não existir avaliação de aluno inexistente, etc.). O uso de um SGBD relacional (RNF04) permite definir essas constraints facilmente.

**Atendimento aos requisitos:** Esse modelo suporta os requisitos funcionais aprimorados: permite armazenar os dados de login (usuários), cursos e associações (para RF02, RF08, RF09), perguntas dinâmicas (RF11), avaliações únicas por aluno-professor-período (RF05, garantido por lógica e estrutura), comentários (RF04) e cálculo de médias (RF07, facilmente obtido calculando média de Resposta.Valor filtrando por Docente e Pergunta). A separação de Resposta por pergunta viabiliza até mostrar médias por pergunta no dashboard. Além disso, o modelo é extensível – poderíamos posteriormente adicionar, por exemplo, uma entidade **Turma** se quiséssemos relacionar avaliações a turmas específicas, ou integrar com tabela de disciplinas, mas para o escopo atual de “curso” está adequado.

**Nota sobre anonimato:** No modelo, a anonimidade é mantida não por estrutura (afinal, Avaliação aponta para Aluno), mas sim pela lógica de aplicação – ou seja, quem consulta os resultados (gestor/diretor) não verá o campo AlunoID. Podemos, se desejado, armazenar a informação de quem avaliou apenas para controle administrativo e nunca expor. Outra possível medida seria ter um campo booleano em Avaliação indicando se está finalizada e deve ser contabilizada, para evitar que alterações pós-período afetem resultados.

Em resumo, o diagrama ER acima demonstra que o sistema foi pensado para atender aos requisitos propostos, garantindo a separação das preocupações: *dados de usuários/controle de acesso*, *dados acadêmicos (cursos, docentes, alunos)* e *dados de avaliação (perguntas, respostas, resultados)*. Com esse modelo implementado, podemos passar para a fase de construção, seguindo as prioridades do MVP e assegurando que os pontos críticos (como anonimato e unicidade) sejam atendidos tanto no design quanto na implementação.

**Referências Utilizadas:** Práticas recomendadas de segurança e armazenamento de senhas foram baseadas nas diretrizes OWASP ([Password Storage - OWASP Cheat Sheet Series](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html" \l ":~:text=Hashing%20and%20encryption%20can%20keep,should%20be%20hashed%2C%20NOT%20encrypted)). A importância de manter o anonimato dos alunos nas avaliações foi destacada em estudos de sistemas de avaliação online ([Developing Online Course and Lecturer Evaluation System using DBase](https://utpedia.utp.edu.my/3933/1/13995_IZYAN_IZZATI_BAHARUDDIN_dissertation_5562.pdf" \l ":~:text=universities,a%20mechanism%20for%20report%20generation)), garantindo feedback honesto sem retaliações. Essas referências e a experiência em sistemas semelhantes embasaram as melhorias propostas. Cada requisito aprimorado busca alinhar o sistema com objetivos de **confiabilidade, usabilidade e integridade**, fundamentais para o sucesso de uma aplicação de Avaliação Docente.