# Atividade avaliativa

**Atenção:** A atividade deverá ser realizada em dupla, caso contrário não será aceita.

## Escopo

Essa atividade avaliativa consiste em implementar um software para acesso de usuários ao sistema, descrito no texto a seguir, cujo Diagrama de Casos de Uso correspondente é apresentado na Figura 1.

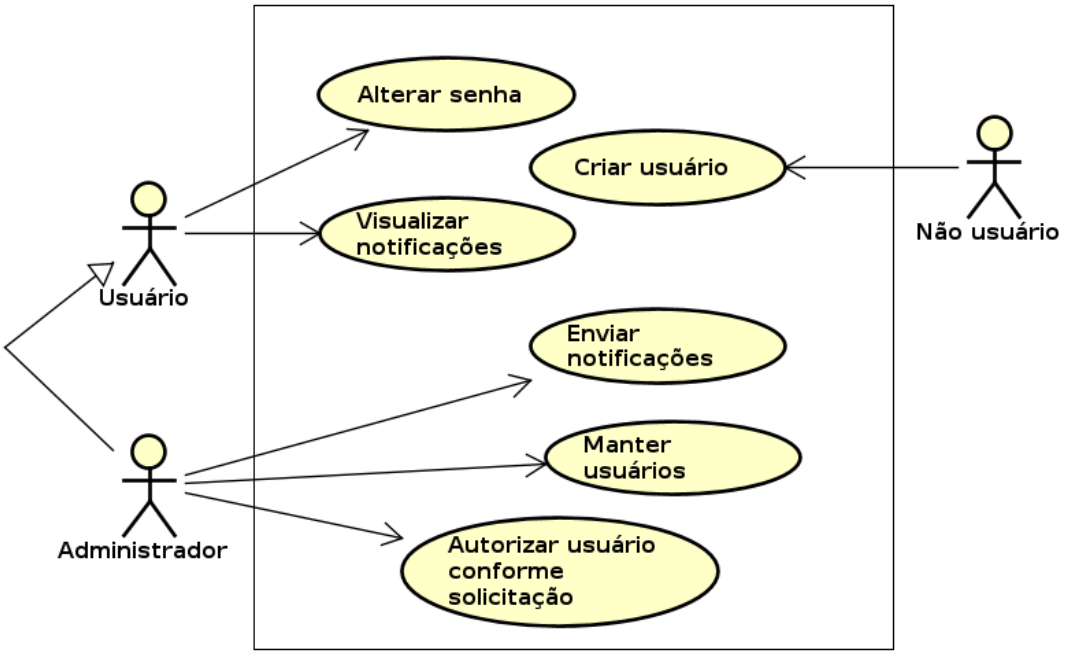


**Figura 1 - Diagrama de Casos de Uso do inicial**

O funcionamento do sistema representado no Diagrama de Casos de Uso da Figura 1 é descrito a seguir. Ao acessar o sistema pela primeira vez, o usuário inicial realizará o próprio cadastro e passará a ser o administrador do sistema, a partir desse momento ele poderá cadastrar outros usuários diretamente. Usuários ainda não cadastrados, aqui denominados “Não usuários”, podem acionar a funcionalidade “Criar usuário” para registrar seus dados, porém o acesso ao sistema somente será liberado após a autorização explícita do administrador.

O Administrador (o primeiro a que se cadastrou) pode promover outros usuários como administradores com as mesmas autorizações que o primeiro administrador, exceto excluir qualquer outro administrador. Apenas o primeiro administrador poderá fazer excluir qualquer outro administrador (ou qualquer usuário) e mudar o perfil de um usuário comum para administrador e vice-versa, exceto fazer essa mudança no seu próprio perfil.

Todos os usuários que já tiverem sido autorizados devem realizar autenticação informando nome de usuário e senha para acesso às funcionalidades disponíveis no sistema.

Os administradores podem enviar mensagens ou notificações para um ou mais usuários cadastrados no sistema, utilizando a funcionalidade de envio de notificações.

Quando acessam o sistema, os usuários devem possuir um meio de visualizar as notificações recebidas, bem como marcá-las como “lida” após a leitura.

Ao listar os usuários, o administrador deve visualizar, para cada usuário, o nome, a data de cadastro, o número total de notificações enviadas e o número de notificações lidas, mantendo assim uma visão quantitativa das interações realizadas via notificações.

Enquanto estiverem autenticados, os usuários podem, a qualquer momento, utilizar a funcionalidade de alteração de senha para atualizar sua credencial de acesso, sempre vinculando a operação ao usuário atualmente autenticado.

O sistema deve registrar em arquivo de log todas as seguintes operações: inclusão, alteração e exclusão de usuários, envio de notificações, leitura de notificações, alteração de senha e autorização de usuário. As mensagens gravadas no arquivo de log devem seguir um formato padronizado. Para operações bem-sucedidas, o formato base é:

```
<OPERACAO>: <NOME>, (<DATA>, <HORA>, <USUARIO>)
```

Para operações que resultarem em falha, o formato base é:

```
Ocorreu a falha <MENSAGEM> ao realizar a operação <OPERACAO> para o usuário <NOME>, (<DATA>, <HORA>, <USUARIO>).
```

Em ambos os casos, as variáveis entre colchetes angulares devem ser substituídas pelos respectivos dados em tempo de execução, onde <NOME> corresponde ao nome do usuário sobre o qual a operação está sendo

realizada, <DATA> corresponde à data atual do sistema operacional no formato DD/MM/AAAA, <HORA> corresponde ao horário atual do sistema operacional no formato HH:MM:SS, <USUARIO> corresponde ao usuário autenticado que executou a operação, <MENSAGEM> corresponde à mensagem de falha retornada pelo sistema e <OPERACAO> corresponde a uma das operações listadas anteriormente.

Exemplos de registros possíveis no log são:

```
INCLUSAO_USUARIO: Ana Souza, (23/11/2025, 14:32:10, administrador)

LEITURA_NOTIFICACAO: Carlos Lima, (23/11/2025, 15:05:42, usuario_padrao)

Ocorreu a falha "Senha em formato inválido" ao realizar a operação ALTERACAO_SENHA para o usuário Joao Silva, (23/11/2025, 15:18:03, usuario_padrao).
```

Deve ser disponibilizada uma funcionalidade de “Configuração” que permita selecionar, dentre os formatos de arquivos de log suportados, aquele que será utilizado pela aplicação, de forma que o usuário autorizado possa alterar o tipo de log em tempo de execução, e a escolha realizada deve ser persistida para uso em acessos posteriores.

No rodapé do sistema devem ser exibidas as seguintes informações: o nome do usuário atualmente autenticado, o tipo associado a esse usuário (Usuário ou Administrador) e um botão que, quando existirem notificações pendentes, deve apresentar como texto o número total de notificações não lidas. Ao acionar esse botão, deve ser aberta uma janela em que o usuário possa visualizar as notificações correspondentes e acessar o que estiver indicado em cada uma delas. Funcionalidades não compatíveis com o perfil do usuário, ou seja, específicas do administrador, não devem aparecer na interface gráfica de usuários não administradores. As User stories fornecem detalhes sobre o funcionamento do sistema.

## Requisitos não funcionais

**RNF01 – Arquitetura Model-View-Presenter (MVP) – Passive View -** Toda a aplicação deve adotar o padrão Model-View-Presenter (MVP) na abordagem Passive View, conforme discutido e implementado em sala de aula, de forma que as classes correspondentes às telas da interface gráfica utilizem obrigatoriamente o sufixo View em seus nomes e as classes responsáveis pela lógica de apresentação utilizem obrigatoriamente o sufixo Presenter, enquanto as classes de modelo de domínio (Model) não devem adotar sufixos específicos. A interação entre Model, View e Presenter deve seguir as responsabilidades apresentadas na disciplina,

mantendo a lógica de apresentação concentrada nas classes Presenter, e restringindo as classes View ao gerenciamento da interface gráfica e à exposição de métodos para leitura e escrita do estado dos componentes, de forma que a sincronização entre Model e View seja realizada pelas classes Presenter.

**RNF02 – Criação de formulários Java Swing -** Todos os formulários da interface gráfica devem ser implementados utilizando o editor visual do NetBeans, de forma que cada formulário Swing possua o respectivo arquivo com extensão ".form". O arquivo ".form" é criado e atualizado automaticamente pelo NetBeans sempre que o desenvolvedor altera o formulário pelo editor visual, descreve a estrutura da tela, os componentes, o layout e as propriedades associadas, e é utilizado pela IDE para reconstruir o formulário e gerar o código Java correspondente, não devendo ser modificado manualmente. Os prefixos dos nomes dos componentes Swing devem usar a definição apresentada em Apêndice A - Prefixos dos componentes Swing.

**RNF03 – Uso de dependências -** Todas as bibliotecas externas utilizadas no projeto devem ser incluídas exclusivamente na seção <dependencies> do arquivo pom.xml do Maven, não sendo permitido o uso de bibliotecas adicionadas manualmente ao classpath ou por qualquer outro mecanismo de inclusão externa fora da gestão de dependências do Maven.

**RNF04 – Persistência de dados -** A persistência de dados deve ser implementada exclusivamente com o banco de dados SQLite, sendo que o arquivo da base de dados, com extensão .db, deve estar localizado na pasta raiz da aplicação. Exemplo de código para conexão, criação da base de dados, inserção, alteração, consulta e remoção de dados utilizando SQLite foi disponibilizado no Google Sala de Aula da disciplina, no endereço: https://classroom.google.com/c/NzgwNjYwNTU5NzYz/m/NzkyNDI5NTkxOTUw/details . Não é permitido utilizar Docker para manter o banco de dados.

**RNF05 – Classe Repository -** A camada de acesso a dados deve ser implementada por meio de classes do tipo Repository, sem utilização de frameworks de persistência como Hibernate ou similares; essas classes devem estar aderentes aos princípios S.O.L.I.D., com ênfase no Princípio da Inversão de Dependência (Dependency Inversion Principle – DIP), para manter baixo acoplamento entre a lógica de domínio e os mecanismos de persistência de dados.

**RNF06 – Interface gráfica MDI (Multiple Document Interface)** - O sistema deve adotar uma interface gráfica do tipo MDI (Multiple Document Interface[1]), permitindo que usuários naveguem entre múltiplas telas abertas para realizar as funções desejadas. Um exemplo de sistema com o padrão MDI é apresentado na Figura 2, sendo a barra de botões um elemento opcional.

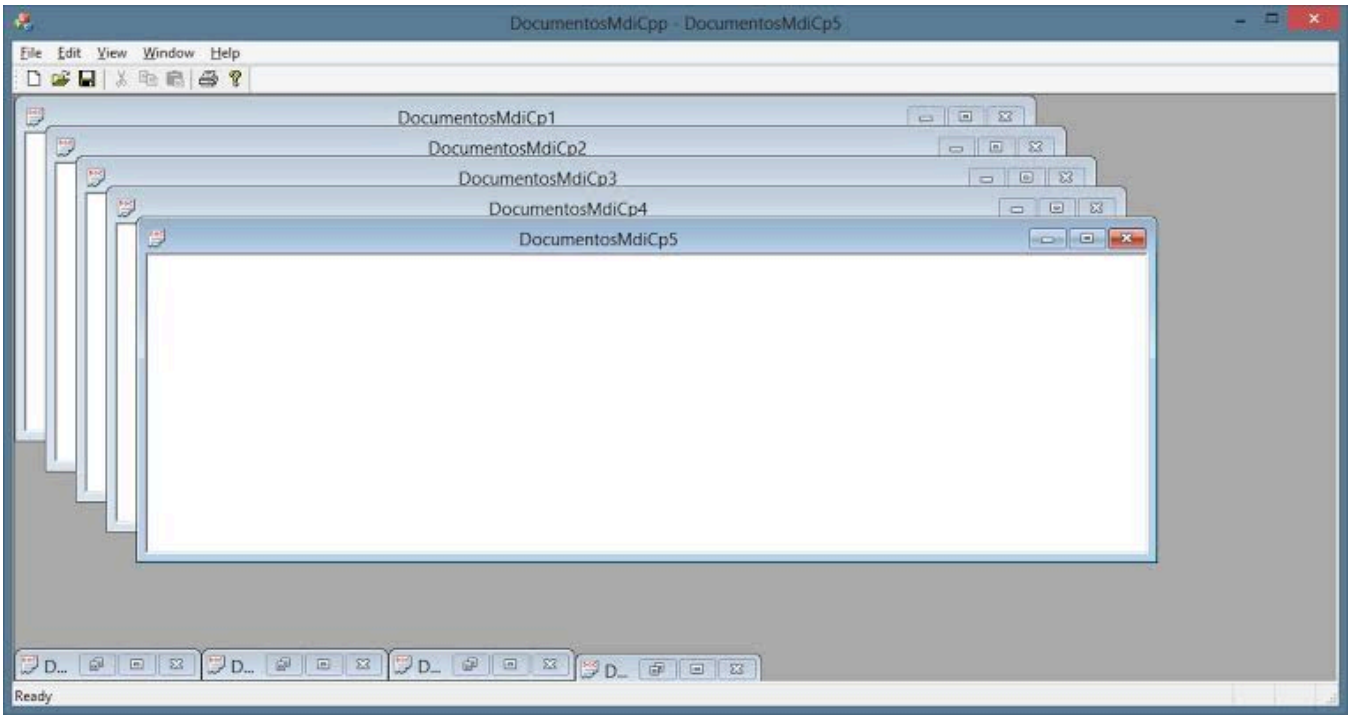


**Figura 2 - Exemplo de janelas com MDI**

**RNF07 – Formatos de arquivos de Log** - O sistema deve gerar arquivos de log nos formatos CSV, com separação de campos utilizando ponto e vírgula (";"), e JSONL (JSON Lines), em que cada linha do arquivo representa um registro de log em formato JSON, mantendo todos os registros de log em um único arquivo, independentemente da data em que foram gerados. O mecanismo de geração e escrita desses arquivos de log deve ser implementado pela equipe como uma biblioteca externa, disponibilizada no JitPack (https://jitpack.io/) para uso na aplicação aqui proposta como dependência, não devendo o código-fonte desse mecanismo de log integrar diretamente o código da aplicação.

[1] https://pt.wikipedia.org/wiki/Interface_de_documentos_m%C3%BAltiplos

**RNF08 – Criação de senhas -** Ao criar senhas do sistema deve ser utilizado o Validador de senhas, disponível em https://github.com/claytonfraga/validadorsenha, que deve ser incluído como dependência[2] no projeto Maven, utilizando o JitPack (https://jitpack.io/).

**RNF09 – Registro de requisitos não implementados -** A dupla deve manter no arquivo README.md do projeto o link para um documento no Google Docs descrevendo explicitamente quais requisitos não foram implementados (requisitos parcialmente implementados são considerados não implementados para todos os efeitos) na aplicação. Caso este requisito não seja atendido a nota da atividade será zerada. Da mesma forma, caso a implementação não contenha algum requisito não declarado aqui, a nota será reduzida na proporção já definida.

**RNF10 – Integridade de compilação -** O projeto entregue não deve apresentar erros de código que impeçam a compilação no ambiente padrão de correção, e projetos que não compilarem nessas condições não serão corrigidos e receberão nota zero.

**RNF11 – Execução da aplicação -** A aplicação entregue deve ser executável no ambiente padrão de correção a partir do projeto fornecido, e projetos que não executarem nessas condições não serão corrigidos e receberão nota zero.

**RNF12 – Aplicação de conceitos da disciplina -** A implementação do sistema deve evidenciar a aplicação dos conceitos e técnicas estudados na disciplina, de forma alinhada ao conteúdo ministrado em aula e ao contexto do problema proposto.

**RNF13 – Aplicação de princípios S.O.L.I.D. e conceitos de projeto -** A implementação do sistema deve aplicar os princípios SOLID e os demais conceitos de projeto e arquitetura de software estudados na disciplina, mantendo essas diretrizes de forma consistente em todas as camadas da aplicação.

**RNF14 – Versão da linguagem Java -** O sistema deve ser desenvolvido utilizando exclusivamente a linguagem Java na versão 17, devendo o projeto estar configurado para compilação e execução nessa versão.

[2] https://www.reviversoft.com/pt/file-extensions/jar

**RNF15 – Uso do Maven como gerenciador de projeto -** O projeto deve ser criado e mantido utilizando o Maven como ferramenta de gerenciamento de projeto e dependências, de forma que a estrutura de diretórios, o arquivo `pom.xml` e o ciclo de build sigam o modelo padrão do Maven.

**RNF16 – Tratamento de exceções -** O tratamento de exceções deve ser implementado em todo o projeto, de modo que operações suscetíveis a falhas sejam envolvidas por mecanismos adequados de captura, tratamento e propagação de exceções, evitando a interrupção não controlada da aplicação e permitindo o registro apropriado de erros.

**RNF17 – Entrega e versionamento no GitHub -** A entrega do projeto deve ser realizada por meio de repositório no GitHub, que deve ser utilizado como repositório de controle de versões e trabalho em equipe, não sendo aceita a entrega apenas como arquivo compactado; o repositório deverá permitir a obtenção do projeto por meio do comando `git clone <link-do-repositorio>`.