



Fundação Presidente Antônio Carlos de
Conselheiro Lafaiete
Engenharia de Computação



**Revisão sistemática de literatura para
desenvolvimento de uma aplicação mobile
focada no aprendizado ubíquo por meio de
repetições espaçadas**

Richard Felipe Ribeiro Reis

Conselheiro Lafaiete, 31 de setembro de 2023

Richard Felipe Ribeiro Reis

**Revisão sistemática de literatura para desenvolvimento de
uma aplicação mobile focada no aprendizado ubíquo por
meio de repetições espaçadas**

Trabalho de conclusão de curso apresentado
como parte das atividades para obtenção do
título de Bacharel em Engenharia de Compu-
tação, da Fundação Presidente Antônio Car-
los de Conselheiro Lafaiete.

Orientador: Jean Carlo Mendes

Conselheiro Lafaiete

31 de setembro de 2023

Lista de ilustrações

Figura 1 – Diagrama Entidade Relacionamento	5
Figura 2 – Diagrama de caso de uso	6
Figura 3 – Tela de autenticação do usuário	7
Figura 4 – Tela de cadastro do usuário	9
Figura 5 – Tela de Listagem dos Baralhos Cadastrados	10
Figura 6 – Tela de cadastro do baralho de estudos	12
Figura 7 – Tela de edição do baralho de estudos	14
Figura 8 – Caixa de dialogo para remoção do baralho de estudos	15
Figura 9 – Tela de gerenciamento do baralho listando suas cartas	16
Figura 10 – Visualização do verso de uma carta dentro da lista	17
Figura 11 – Tela de cadastro de cartas	18
Figura 12 – Tela de edição de carta	20
Figura 13 – Caixa de dialogo para remover uma carta	21
Figura 14 – Tela de revisão de uma carta	23
Figura 15 – Tela de revisão de uma carta ao revelar verso	23
Figura 16 – Caixa de dialogo com desempenho de um baralho	25
Figura 17 – Tela de listagem das salas de estudo	26
Figura 18 – Tela de cadastro de sala de estudo	28
Figura 19 – Tela de edição de uma sala de estudo	29
Figura 20 – Caixa de dialogo para remover uma Sala de estudo	31
Figura 21 – Tela de edição de dados do usuário logado	35
Figura 22 – Tela de apagar conta do usuário	36
Figura 23 – Tela para encerrar sessão do usuário	37

1 Desenvolvimento

Nesta seção, discutiremos as táticas fundamentais que levaram ao nosso processo de desenvolvimento do aplicativo. Discutiremos a arquitetura selecionada, os algoritmos implementados, o levantamento de requisitos realizado, a modelagem do banco de dados e apresentaremos o diagrama de casos de uso, acompanhado do detalhamento das decisões e escolhas que levaram à criação da aplicação, bem como na solução final que apresentamos na conclusão do projeto.

1.1 Aplicativo

O processo de desenvolvimento da aplicação mobile baseada em repetições espaçadas para o aprendizado ubíquo foi guiado pelo uso de diferentes linguagens de programação, ferramentas e metodologias. Para a construção da interface da aplicação foi utilizado a biblioteca JavaScript React Native. Além disso, foram aplicados conceitos de design thinking para o desenvolvimento da experiência do usuário.

A criação da aplicação mobile baseada em repetições espaçadas para o aprendizado ubíquo envolveu a consideração de diversos fatores, como a seleção dos algoritmos de repetições espaçadas mais adequados para a aplicação, a definição dos recursos e funcionalidades necessários para a aplicação, e a definição das estratégias de personalização da experiência de aprendizado do usuário.

A aplicação mobile também utilizou conceitos de gamificação, com a utilização de elementos como o sistema de pontuação de acordo com a quantidade de estudos revisados diariamente, para dar ao usuário uma sensação de progressão. Além disso, foram aplicados conceitos de aprendizado colaborativo, com a possibilidade de compartilhamento de baralhos com outros usuários e a criação de salas de estudo.

No desenvolvimento também tentamos focar em conceitos de acessibilidade, com a utilização de recursos de navegação por voz e a possibilidade de adaptação do tamanho da fonte e do contraste da tela. Contudo por se tratar de um protótipo essas funcionalidades foram desenvolvidas de forma mais simples, tendo ainda uma grande margem para evoluir e tornar o aplicativo mais abrangente.

Para atingir o objetivo foi utilizado no processo de desenvolvimento do algoritmo de repetição espaçada, uma integração com uma aplicação de código aberto conhecida como Ateno para a construção do nosso back-end. Para isso foi desenvolvida uma API em Asp Net Core que pudesse conversar com as camadas de negócio da aplicação e com nosso aplicativo mobile. Dessa forma a arquitetura do back end ficou da seguinte forma:

Essa escolha permitiu que o desenvolvimento aproveitasse varios dos benefícios da aplicação, bem como aprimorasse a escalabilidade e a eficiência de nosso back-end. A integração com o Ateno também nos proporcionou uma sólida base para a gestão de dados, autenticação de usuários e a execução de operações essenciais para o funcionamento do aplicativo.

1.2 Requisitos

1.2.1 Requisitos Funcionais

- O sistema deve permitir usuarios cadastrados iniciem uma sessão.
- O sistema deve permitir usuarios realizem o cadastro no aplicativo.
- O sistema deve permitir que o usuario crie novos baralhos de estudo.
- O sistema deve permitir que o usuario edite baralhos de estudo.
- O sistema deve permitir que o usuario delete baralhos de estudo.
- O sistema deve permitir que o usuario tenha acesso aos baralhos de estudo ja criados.
- O sistema deve permitir que o usuário crie cartas e associe a um baralho de estudo ja criado.
- O sistema deve permitir que o usuário edite cartas já associadas a um baralho.
- O sistema deve permitir que o usuário delete cartas já associadas a um baralho.
- O sistema deve permitir que o usuário acesse todas as cartas associadas de um baralho.
- O sistema deve permitir que o usuário inicie um estudo a partir de um baralho.
- O sistema deve permitir que apenas baralhos com no minimo uma carta estejam disponiveis para estudo.
- O sistema deve permitir que o usuário acesse seu desempenho em cada baralho de estudo individualmente.
- O sistema deve permitir que o usuário crie salas de estudos.
- O sistema deve permitir que o usuário marque suas salas de estudos como privadas.
- O sistema deve permitir que apenas usuários com permissões acessem salas de estudo privadas.

- O sistema deve permitir que todos os usuários acessem salas de estudos marcadas como públicas.
- O sistema deve permitir que o usuário que criou uma sala de estudo privada adicione outros usuarios.
- O sistema deve permitir que o usuário que criou uma sala de estudo privada remova usuarios da sala.
- O sistema deve permitir que o usuário edite salas de estudos.
- O sistema deve permitir que o usuário edite salas de estudos.
- O sistema deve permitir que o usuário delete salas de estudos.
- O sistema deve permitir que o usuario tenha acesso as salas de estudo a qual é vinculado.
- O sistema deve permitir que o usuario tenha acesso aos baralhos associados a sala de estudo.
- O sistema deve permitir que o usuário faça a edição dos seus dados de conta.
- O sistema deve permitir que usuario delete sua conta.
- O sistema deve permitir que o usuário encerre sua sessão.

1.2.2 Requisitos Não Funcionais

- O sistema deve permitir o acesso ao sistema apenas por usuarios cadastrados.
- O sistema deve permitir que o usuario tenha acesso apenas aos baralhos criados pelo mesmo.
- O sistema deve permitir que apenas o usuário que criou a baralho realize a edição do mesmo.
- O sistema deve permitir que apenas o usuário que criou a baralho realize a deleção do mesmo.
- O sistema deve permitir que os usuarios tenham acesso apenas as salas de estudos criadas pelo mesmo ou publicas.
- O sistema deve permitir que o usuario tenha acesso a salas de estudo privadas apenas quando vinculados pelo usuario dono da sala.

-
- O sistema deve permitir que apenas o usuário que criou a sala de estudo realize a edição da mesma.
 - O sistema deve permitir que apenas o usuário que criou a sala de estudo realize a delegação da mesma.
 - O sistema deve permitir que o usuário mantenha sua sessão ativa mesmo ao sair do aplicativo.
 - O sistema deve permitir que o usuário delete sua conta apenas após iniciar uma sessão.

1.3 Diagrama Entidade Relacionamento

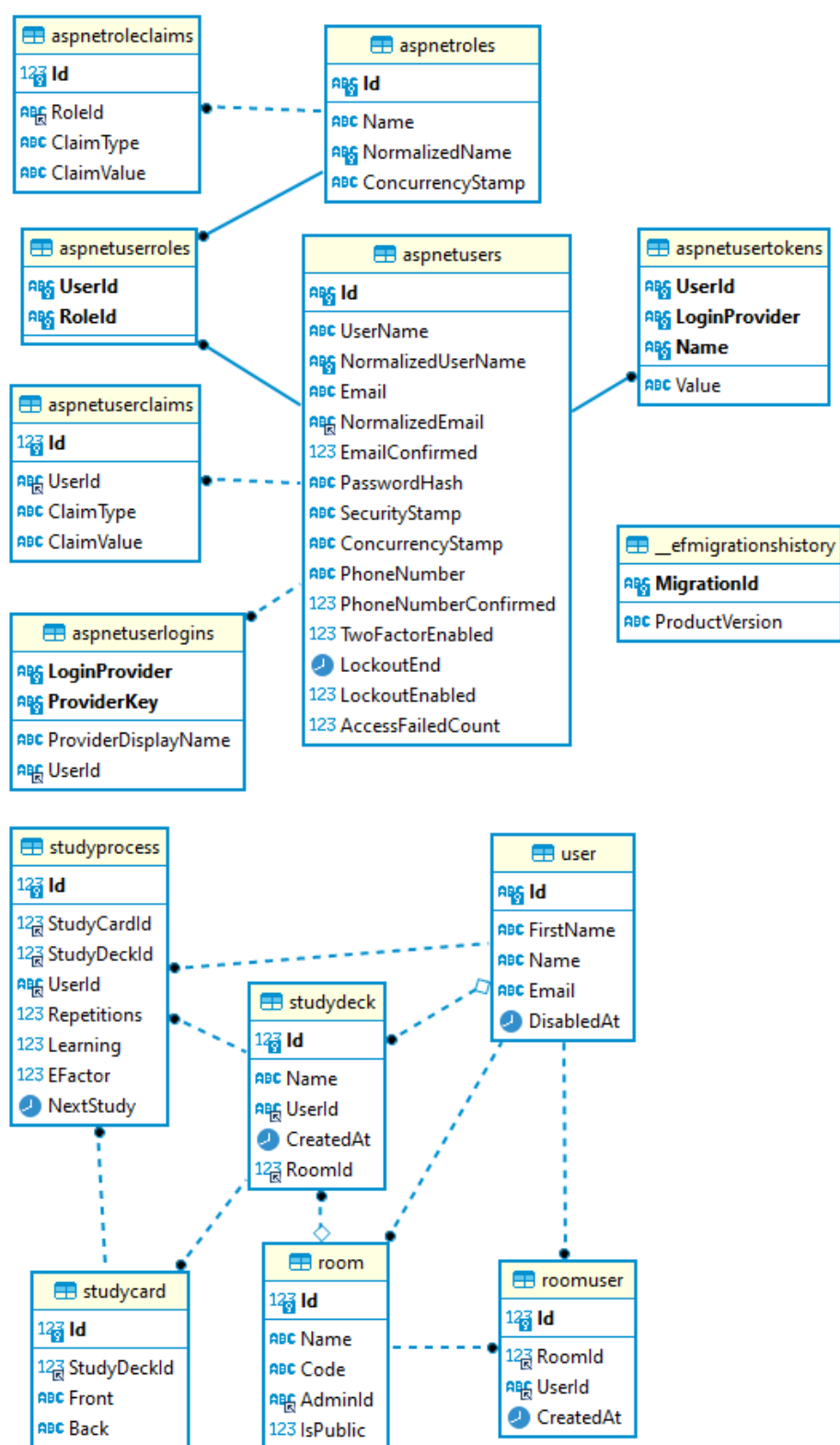


Fig. 1. Diagrama Entidade Relacionamento

1.4 Caso de Uso

1.4.1 Diagrama de caso de uso

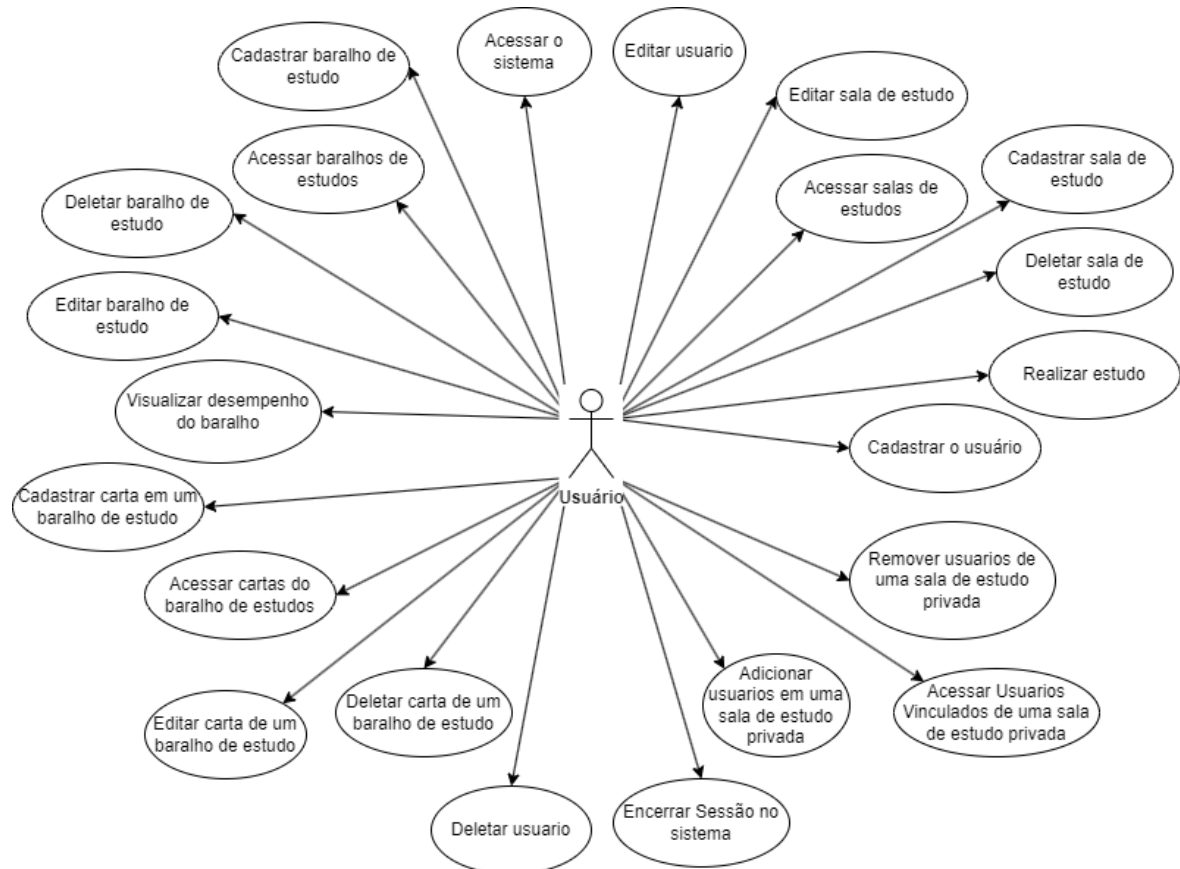


Fig. 2. Diagrama de caso de uso

1.4.2 Detalhamento dos casos de uso

1.4.2.1 UC0001: Acessar o sistema

- **Requisito:** O sistema deve permitir usuarios cadastrados iniciem uma sessão.
- **Objetivo:** Autenticar e validar o usuario para liberar seu acesso ao aplicativo.
- **Descrição:** Na tela inicial do aplicativo haverá um formulário onde o usuario deverá informar email e senha previamente cadastrados para validar seu acesso ao aplicativo.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário já deve estar cadastrado no sistema.
- **Pós-Condição:** O usuario deve ser informado sobre o resultado da validação do seu usuario e em caso de sucesso o aplicativo deve redirecionar para tela principal.

- **Fluxos de eventos:**

- **Fluxo principal:**

1. O usuario acessa a tela principal do aplicativo;
2. O usuário deve informar seu Email e Senha nos campos corretos do formulario;
3. O usuário deve enviar o formulario;
4. O aplicativo deve enviar uma requisição para API;
5. O aplicativo deve receber a resposta da requisição e valida-la;
6. O aplicativo deve salvar o token de autenticação do usuario caso a resposta da API apresente um sucesso;
7. O aplicativo de redirecionar o usuario para tela principal.
8. Caso de uso encerrado.

- **Fluxo alternativo:**

4. O aplicativo deve informar ao usuario caso a resposta da API apresente algum problema da validação;
5. Caso de uso encerrado.

- **Requisitos de tela:**

A fig. 3 apresenta o formulário de *login*.



Fig. 3. Tela de autenticação do usuário

1.4.2.2 UC0002: Cadastrar o usuário

- **Requisito:** O sistema deve permitir usuarios realizem o cadastro no aplicativo.

- **Objetivo:** Permitir que um usuario realize seu cadastro para utilizar o aplicativo.
- **Descrição:** Uma tela que possui um formulario para que os Usuarios informem os dados necessarios para que o mesmo se cadastre.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario não possuir uma conta ativa.
- **Pós-Condição:** Ao validar todos os dados o usuario deve acessar automaticamente o aplicativo.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario acessa a tela de Cadastro;
 2. O usuário deve informar todos os dados obrigatorios nos campos corretos do formulario;
 3. O usuário deve enviar o formulario;
 4. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
 5. O aplicativo deve enviar uma requisição para API;
 6. O aplicativo deve receber a resposta da requisição e valida-la;
 7. O aplicativo deve realizar o Login do usuario de forma automatica;
 8. O aplicativo de redirecionar o usuario para tela principal;
 9. Caso de uso encerrado.
 - **Fluxo alternativo:**
 7. O aplicativo deve informar ao usuario caso a resposta da API apresente algum problema no cadastro;
 8. Caso de uso encerrado.

- **Requisitos de tela:**

A fig. 4 apresenta a tela onde o usuário pode realizar seu cadastro no sistema.

20:11 [ícones] 90%

Cadastre-se

Retenha o conhecimento de forma simples e de um boot na sua vida acadêmica!

Nome

E-mail

Senha

Cadastrar

Já possui uma conta? [Entrar](#)

[Barra de navegação inferior com ícones de menu, home e voltar]

Fig. 4. Tela de cadastro do usuário

1.4.2.3 UC0003: Acessar baralhos de estudos

- **Requisito:** O sistema deve permitir que o usuário tenha acesso aos baralhos de estudo já criados.
- **Objetivo:** Permitir que o usuário possa acessar todos os baralhos de estudos que ele possui.
- **Descrição:** Uma tela que lista todos os baralhos de estudos do usuário logado, além de acesso a botões para manipulá-los.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**

1. O usuário acessa a tela de Baralhos;
2. O aplicativo deve enviar uma requisição para API;
3. O aplicativo deve receber a resposta da requisição e validá-la;
4. O aplicativo deve listar todos os baralhos presentes na resposta da API;
5. Caso de uso encerrado;

– **Fluxo alternativo:**

4. O aplicativo informa que não há baralhos cadastrados e exibe um botão para realizar o cadastro;
5. Caso de uso encerrado;

- **Requisitos de tela:**

A fig. 5 apresenta a tela com a listagem dos baralhos do usuário.



Fig. 5. Tela de Listagem dos Baralhos Cadastrados

1.4.2.4 UC0004: Cadastrar baralho de estudo

- **Requisito:** O sistema deve permitir que o usuário crie novos baralhos de estudo.
- **Objetivo:** Permitir que o usuário realize o cadastro de um novo baralho de estudo.
- **Descrição:** Uma tela que possui um formulário para que o usuário cadastre novos baralhos individuais ou dentro de uma sala de estudos.

- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario acessa a tela de Cadastro de Baralho;
 2. O usuário deve informar todos os dados obrigatorios nos campos corretos do formulario;
 3. O usuário deve enviar o formulario;
 4. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
 5. O aplicativo deve enviar uma requisição para API;
 6. A API realiza o cadastro do baralho e retorna o resultado para a aplicação;
 7. O aplicativo deve receber a resposta da requisição e valida-la;
 8. O aplicativo redireciona o usuario para a Lista de Baralhos atualizada;
 9. Caso de uso encerrado.
 - **Fluxo alternativo:**
 6. O aplicativo deve informar em tela ao usuario caso a resposta da API apresente algum problema no cadastro;
 7. Caso de uso encerrado.
 - **Fluxo alternativo:**
 1. O usuario dono da Sala de Estudos acessa a tela de Cadastro de Baralho;
 2. O usuário deve informar todos os dados obrigatorios nos campos corretos do formulario;
 3. O usuário deve enviar o formulario;
 4. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
 5. O aplicativo deve enviar uma requisição para API;
 6. A API realiza o cadastro do baralho e retorna o resultado para a aplicação;
 7. O aplicativo deve receber a resposta da requisição e valida-la;
 8. O aplicativo redireciona o usuario para a Lista de Baralhos da Sala de Estudos atualizada;
 9. Caso de uso encerrado.
- **Requisitos de tela:**

A fig. 6 apresenta a tela onde o usuário pode realizar o cadastro de um novo baralho.

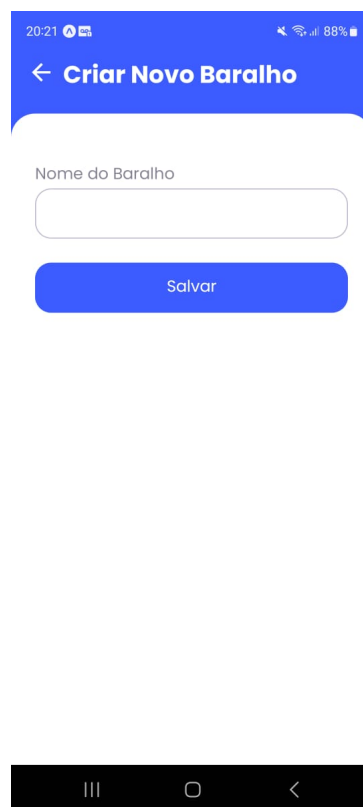


Fig. 6. Tela de cadastro do baralho de estudos

1.4.2.5 UC0005: Editar baralho de estudo

- **Requisito:** O sistema deve permitir que o usuário edite baralhos de estudo.
- **Objetivo:** Permitir que o usuário realize alterações em um baralho de estudo.
- **Descrição:** Uma tela que possui um formulário para que o usuário possa editar os dados de um baralho individual ou dentro de uma sala de estudos.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário acessa a tela de Editar Baralho;
 2. O aplicativo deve enviar uma requisição para API para buscar os dados do baralho;
 3. O aplicativo deve preencher os campos com os dados do baralho;
 4. O usuário deve informar todos os dados obrigatórios nos campos corretos do formulário;

5. O usuário deve enviar o formulário;
6. O aplicativo deve validar se todos os campos obrigatórios foram preenchidos;
7. O aplicativo deve enviar uma requisição para API;
8. A API realiza a edição do baralho e retorna o resultado para a aplicação;
9. O aplicativo deve receber a resposta da requisição e validá-la;
10. O aplicativo redireciona o usuário para a Lista de Baralhos atualizada;
11. Caso de uso encerrado.

– **Fluxo alternativo:**

8. O aplicativo deve informar em tela ao usuário caso a resposta da API apresente algum problema na edição;
9. Caso de uso encerrado.

– **Fluxo alternativo:**

1. O usuário dono da Sala de Estudos acessa a tela de Editar Baralho;
2. O aplicativo deve enviar uma requisição para API para buscar os dados do baralho;
3. O aplicativo deve preencher os campos com os dados do baralho;
4. O usuário deve informar todos os dados obrigatórios nos campos corretos do formulário;
5. O usuário deve enviar o formulário;
6. O aplicativo deve validar se todos os campos obrigatórios foram preenchidos;
7. O aplicativo deve enviar uma requisição para API;
8. A API realiza a edição do baralho e retorna o resultado para a aplicação;
9. O aplicativo deve receber a resposta da requisição e validá-la;
10. O aplicativo redireciona o usuário para a Lista de Baralhos da Sala de Estudos atualizada;
11. Caso de uso encerrado.

• **Requisitos de tela:**

A [fig. 7](#) apresenta a tela onde o usuário pode editar os baralhos de estudos.

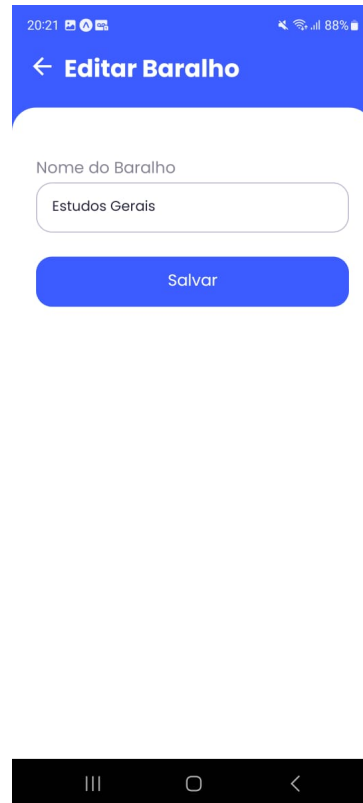


Fig. 7. Tela de edição do baralho de estudos

1.4.2.6 UC0006: Deletar baralho de estudo

- **Requisito:** O sistema deve permitir que o usuário delete baralhos de estudo.
- **Objetivo:** Permitir que o usuário delete um baralho de estudos.
- **Descrição:** Por meio de um botão o usuário poderá deletar baralhos da Lista de Baralhos.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário clica no botão de deletar baralho de estudo;
 2. O aplicativo deve confirmar a intenção do usuário de deletar aquele baralho;
 3. O aplicativo deve enviar uma requisição para API;
 4. A API realiza a deleção do baralho e retorna o resultado para a aplicação;
 5. O aplicativo deve receber a resposta da requisição e validá-la;

6. O aplicativo redireciona o usuário para a Lista de Baralhos atualizada;
7. Caso de uso encerrado.

– **Fluxo alternativo:**

6. O aplicativo deve informar em tela ao usuário caso a resposta da API apresente algum problema ao remover baralho;
7. Caso de uso encerrado.

- **Requisitos de tela:** A fig. 8 apresenta a caixa de diálogo para que o usuário de a confirmação para a exclusão do baralho.



Fig. 8. Caixa de diálogo para remoção do baralho de estudos

1.4.2.7 UC0007: Acessar cartas do baralho de estudos

- **Requisito:** O sistema deve permitir que o usuário acesse todas as cartas associadas de um baralho.
- **Objetivo:** Permitir que o usuário possa acessar todas as cartas associadas em um baralho.
- **Descrição:** Uma tela que lista todas as cartas de um baralho de estudo do usuário logado.
- **Ator:** Usuário.

- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario clica em um baralho e acessa a tela de Cartas;
 2. O aplicativo deve enviar uma requisição para API;
 3. O aplicativo deve receber a resposta da requisição e valida-la;
 4. O aplicativo deve listar todas as cartas do baralho presentes na resposta da API;
 5. Caso de uso encerrado;
 - **Fluxo alternativo:**
 4. O aplicativo informa que não há cartas associadas ao baralho e exibe um botão para criar uma carta;
 5. Caso de uso encerrado;
- **Requisitos de tela:**

A fig. 9 apresenta a tela com a listagem de todas as cartas de uma baralho específico.

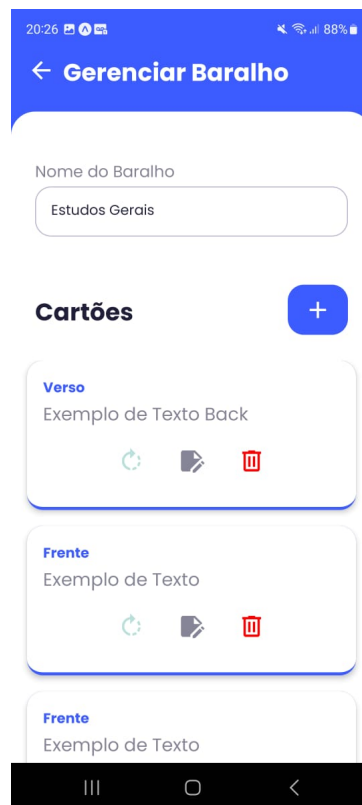


Fig. 9. Tela de gerenciamento do baralho listando suas cartas

A fig. 10 apresenta a tela com a listagem de todas as cartas de uma baralho específico, demonstrando a funcionalidade de visualizar o verso de uma carta.

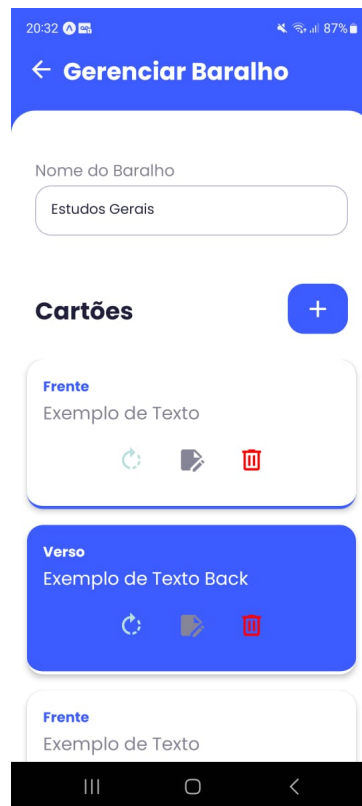


Fig. 10. Visualização do verso de uma carta dentro da lista

1.4.2.8 UC0008: Cadastrar carta em um baralho de estudo

- **Requisito:** O sistema deve permitir que o usuário crie cartas e associe a um baralho de estudo já criado.
- **Objetivo:** Permitir que o usuário realize o cadastro de uma carta em um baralho de estudo.
- **Descrição:** Uma tela que possui um formulário para que o usuário cadastre cartas em um baralho de estudo.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário acessa a tela de Cadastro de Cartas;

2. O usuário deve informar todos os dados obrigatórios nos campos corretos do formulário;
3. O usuário deve enviar o formulário;
4. O aplicativo deve validar se todos os campos obrigatórios foram preenchidos;
5. O aplicativo deve enviar uma requisição para API;
6. A API realiza o cadastro da carta e associa ao baralho de estudo e retorna o resultado para a aplicação;
7. O aplicativo deve receber a resposta da requisição e validá-la;
8. O aplicativo redireciona o usuário para a Lista de Cartas do baralho atualizada;
9. Caso de uso encerrado.

– **Fluxo alternativo:**

6. O aplicativo deve informar em tela ao usuário caso a resposta da API apresente algum problema no cadastro;
7. Caso de uso encerrado.

• **Requisitos de tela:**

A fig. 11 apresenta a tela de cadastro de um novo cartão dentro de um baralho de estudos.



Fig. 11. Tela de cadastro de cartas

1.4.2.9 UC0009: Editar carta de um baralho de estudo

- **Requisito:** O sistema deve permitir que o usuário edite cartas já associadas a um baralho.
- **Objetivo:** Permitir que o usuario realize a edição de uma carta em um baralho de estudo.
- **Descrição:** Uma tela que possui um formulario para que o usuario possa editar cartas em um baralho de estudo.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario acessa a tela de Editar Carta;
 2. O aplicativo deve enviar uma requisição para API para buscar os dados da carta selecionada;
 3. O aplicativo deve preencher os campos com os dados do baralho;
 4. O usuário deve informar todos os dados obrigatorios nos campos corretos do formulario;
 5. O usuário deve enviar o formulario;
 6. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
 7. O aplicativo deve enviar uma requisição para API;
 8. A API realiza a edição da carta e retorna o resultado para a aplicação;
 9. O aplicativo deve receber a resposta da requisição e valida-la;
 10. O aplicativo redireciona o usuario para a Lista de Cartas do baralho atualizada;
 11. Caso de uso encerrado.
 - **Fluxo alternativo:**
 8. O aplicativo deve informar em tela ao usuario caso a resposta da API apresente algum problema na edição;
 9. Caso de uso encerrado.

- **Requisitos de tela:**

A fig. 12 apresenta a tela de edição de uma carta dentro de um baralho de estudos.

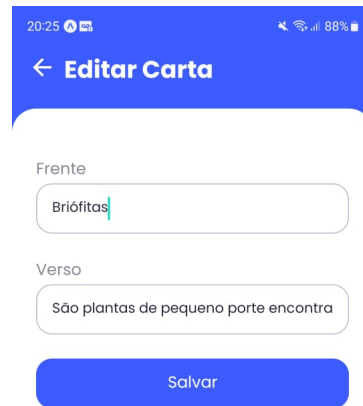


Fig. 12. Tela de edição de carta

1.4.2.10 UC0010: Deletar carta de um baralho de estudo

- **Requisito:** O sistema deve permitir que o usuário delete cartas já associadas a um baralho.
- **Objetivo:** Permitir que o usuário delete uma carta associada a um baralho de estudos.
- **Descrição:** Por meio de um botão o usuário poderá deletar cartas de um baralho de estudo.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**

1. O usuário clica no botão de deletar carta;
2. O aplicativo deve confirmar a intenção do usuário de deletar aquela carta;
3. O aplicativo deve enviar uma requisição para API;
4. A API realiza a deleção da carta selecionada e retorna o resultado para a aplicação;
5. O aplicativo deve receber a resposta da requisição e validá-la;
6. O aplicativo redireciona o usuário para a Lista de Cartas do baralho atualizada;
7. Caso de uso encerrado.

– **Fluxo alternativo:**

6. O aplicativo deve informar em tela ao usuário caso a resposta da API apresente algum problema ao remover carta do baralho;
7. Caso de uso encerrado.

• **Requisitos de tela:**

A fig. 13 apresenta a caixa de diálogo para que o usuário de a confirmação para a exclusão de uma carta.



Fig. 13. Caixa de diálogo para remover uma carta

1.4.2.11 **UC0011:** Realizar estudo

- **Requisito:** O sistema deve permitir que o usuário inicie um estudo a partir de um baralho.

- **Objetivo:** Permitir que o usuario possa utilizar os baralhos cadastrados para estudar.
- **Descrição:** Uma tela onde o usuario tera acesso as cartas cadastradas no baralho de estudo e podera realizar o estudo.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**

- **Fluxo principal:**

1. O usuario clica em um botão do baralho para iniciar estudo;
2. O aplicativo deve enviar uma requisição para API;
3. O aplicativo deve receber a resposta da requisição e valida-la;
4. O aplicativo deve apresentar uma carta para o usuario;
5. O usuario define o nivel de dificuldade para acertar a resposta da carta;
6. O aplicativo passa para proxima carta e repete o processo até passar por todas as cartas;
7. O aplicativo envia todas as respostas para API;
8. A API atualiza o processo de estudo das cartas e atualiza a agenda para o proximo estudo;
9. A API retorna o resultado para a aplicação;
10. O aplicativo redireciona o usuario para a tela de Lista de Baralhos;
11. Caso de uso encerrado;

- **Fluxo alternativo:**

4. O aplicativo informa ao usuario que não há estudos pendentes no baralho selecionado;
5. O aplicativo redireciona o usuario para a tela de Lista de Baralhos;
6. Caso de uso encerrado.

- **Requisitos de tela:**

A fig. 14 apresenta a tela de revisão de uma carta, inicialmente é apresentado apenas o texto da "frente" do cartão.

A fig. 15 apresenta a tela de revisão de uma carta ao revelar o verso da carta, apresentando os botões de feedback.

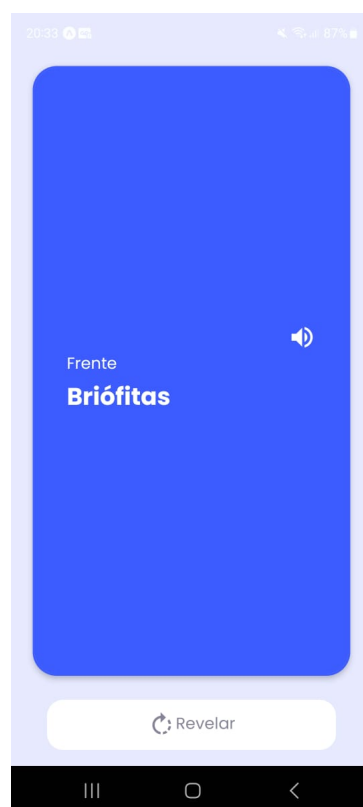


Fig. 14. Tela de revisão de uma carta

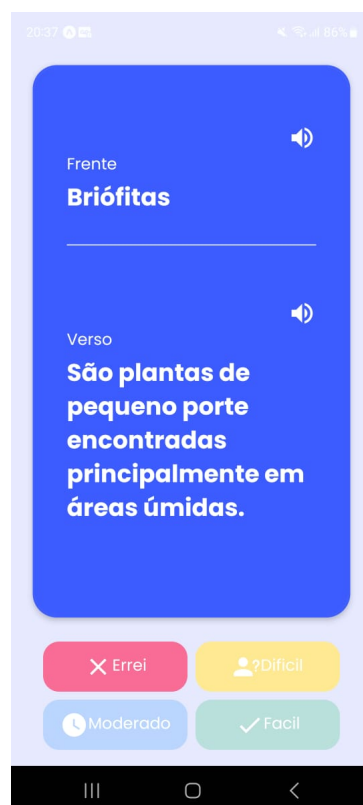


Fig. 15. Tela de revisão de uma carta ao revelar verso

1.4.2.12 UC0012: Visualizar desempenho do baralho

- **Requisito:** O sistema deve permitir que o usuário acesse seu desempenho em cada baralho de estudo individualmente.
- **Objetivo:** Permitir que o usuario possa acompanhar o desempenho de um baralho de estudo.
- **Descrição:** Uma tela onde o usuario tera acesso as metricas de desempenho no baralho selecionado.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario clica em um botão do baralho para visualizar desempenho;
 2. O aplicativo deve enviar uma requisição para API;
 3. O aplicativo recebe a resposta da requisição e valida;
 4. O aplicativo deve mostrar os resultados na tela;
 5. Caso de uso encerrado;
- **Requisitos de tela:**

A fig. 16 apresenta a caixa de dialogo onde o usuário de acesso ao desempenho de um baralho.



Fig. 16. Caixa de diálogo com desempenho de um baralho

1.4.2.13 UC0013: Acessar salas de estudos

- **Requisito:** O sistema deve permitir que o usuário tenha acesso às Salas de Estudos já criadas.
- **Objetivo:** Permitir que o usuário possa acessar todas as salas de estudos às quais ele possui acesso.
- **Descrição:** Uma tela que lista todas as salas de estudos que o usuário logado é dono ou possui permissão para entrar, além de acesso a botões para manipulá-las.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário acessa a tela de Salas de Estudo;
 2. O aplicativo deve enviar uma requisição para a API;
 3. O aplicativo deve receber a resposta da requisição e validá-la;

4. O aplicativo deve listar todas as salas de estudos presentes na resposta da API;
5. Caso de uso encerrado;

– **Fluxo alternativo:**

4. O aplicativo informa que o usuário não está vinculado a nenhuma sala de estudo e exibe um botão para realizar o cadastro;
5. Caso de uso encerrado;

- **Requisitos de tela:**

A fig. 17 apresenta a tela onde o usuário pode acessar a listagem de salas em que ele possui acesso.

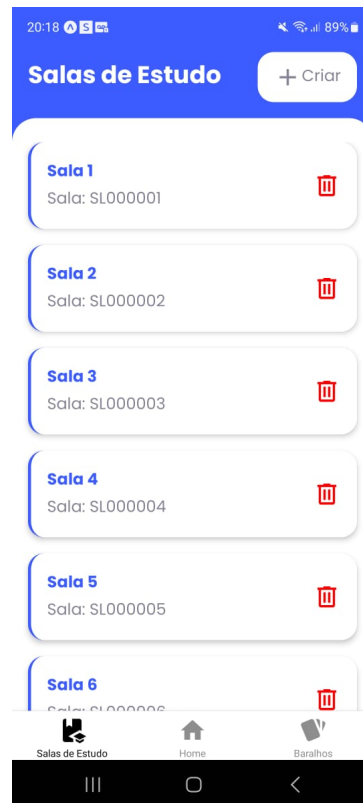


Fig. 17. Tela de listagem das salas de estudo

1.4.2.14 UC0014: Cadastrar sala de estudo

- **Requisito:** O sistema deve permitir que o usuário crie salas de estudos.
- **Objetivo:** Permitir que o usuário realize o cadastro de uma nova sala de estudo.
- **Descrição:** Uma tela que possui um formulário para que o usuário cadastre novas salas de estudos.
- **Ator:** Usuário.

- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario acessa a tela de Cadastro de Sala de Estudo;
 2. O usuário deve informar todos os dados obrigatorios nos campos corretos do formulario;
 3. O usuário deve enviar o formulario;
 4. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
 5. O aplicativo deve enviar uma requisição para API;
 6. A API realiza o cadastro da sala de estudo e retorna o resultado para a aplicação;
 7. O aplicativo deve receber a resposta da requisição e valida-la;
 8. O aplicativo redireciona o usuario para a tela da Sala de Estudo;
 9. Caso de uso encerrado.
 - **Fluxo alternativo:**
 6. O aplicativo deve informar em tela ao usuario caso a resposta da API apresente algum problema no cadastro;
 7. Caso de uso encerrado.
- **Requisitos de tela:**

A fig. 18 apresenta a tela onde o usuário pode realizar o cadastro de uma nova sala de estudo.

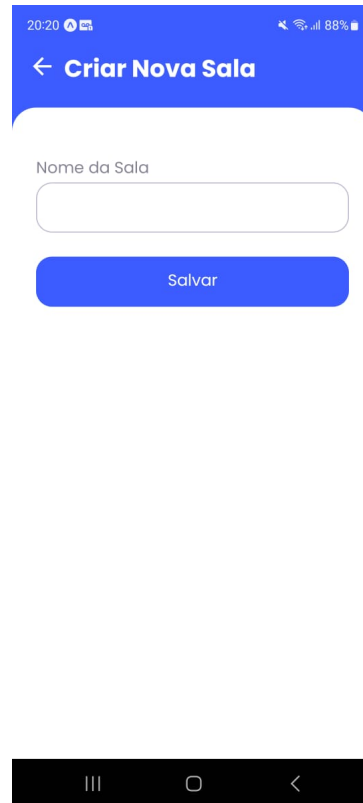


Fig. 18. Tela de cadastro de sala de estudo

1.4.2.15 UC0015: Editar sala de estudo

- **Requisito:** O sistema deve permitir que o usuário edite salas de estudos.
- **Objetivo:** Permitir que o usuário realize alterações em uma sala de estudo.
- **Descrição:** Uma tela que possui um formulário para que o usuário possa editar os dados de uma sala de estudos.
- **Ator:** Usuário.
- **Pré-Condição:** O usuário deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário acessa a tela de Editar Sala de Estudo;
 2. O aplicativo deve enviar uma requisição para API para buscar os dados da sala de estudo;
 3. O aplicativo deve preencher os campos com os dados da sala de estudo;

4. O usuário deve informar todos os dados obrigatórios nos campos corretos do formulário;
5. O usuário deve enviar o formulário;
6. O aplicativo deve validar se todos os campos obrigatórios foram preenchidos;
7. O aplicativo deve enviar uma requisição para API;
8. A API realiza a edição do baralho e retorna o resultado para a aplicação;
9. O aplicativo deve receber a resposta da requisição e validá-la;
10. O aplicativo redireciona o usuário para a tela da Sala de Estudo;
11. Caso de uso encerrado.

– **Fluxo alternativo:**

8. O aplicativo deve informar em tela ao usuário caso a resposta da API apresente algum problema na edição;
9. Caso de uso encerrado.

• **Requisitos de tela:**

A fig. 19 apresenta a tela onde o usuário pode realizar a edição de uma sala de estudo.

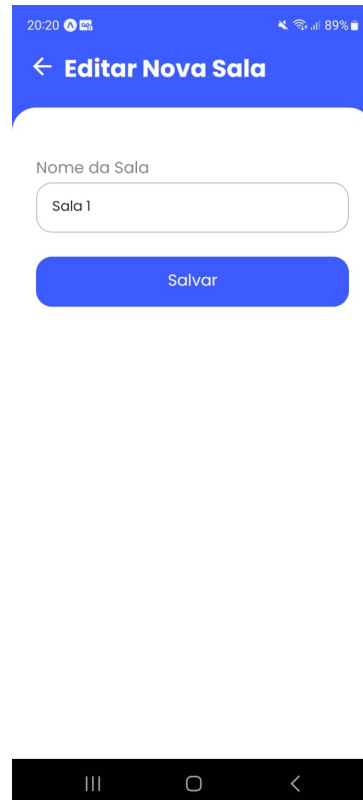


Fig. 19. Tela de edição de uma sala de estudo

1.4.2.16 UC0016: Deletar sala de estudo

- **Requisito:** O sistema deve permitir que o usuário delete salas de estudos.
- **Objetivo:** Permitir que o usuario delete uma sala de estudos.
- **Descrição:** Por meio de um botão o usuario poderá deletar uma sala de estudo.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário clica no botão de deletar sala de estudo;
 2. O aplicativo deve confirmar a intenção do usuario de deletar aquela sala de estudo;
 3. O aplicativo deve enviar uma requisição para API;
 4. A API realiza a deleta a sala de estudo e retorna o resultado para a aplicação;
 5. O aplicativo deve receber a resposta da requisição e valida-la;
 6. O aplicativo redireciona o usuario para a Lista de Salas de Estudos atualizada;
 7. Caso de uso encerrado.
 - **Fluxo alternativo:**
 6. O aplicativo deve informar em tela ao usuario caso a resposta da API apresente algum problema ao remover a sala de estudo;
 7. Caso de uso encerrado.
- **Requisitos de tela:**

A fig. 20 apresenta a caixa de dialogo onde o usuário pode confirmar a exclusão de uma sala de estudo.

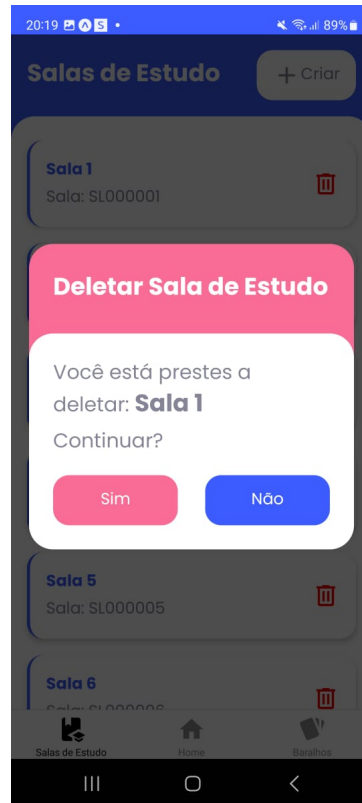


Fig. 20. Caixa de dialogo para remover uma Sala de estudo

1.4.2.17 UC0017: Acessar Usuarios Vinculados de uma sala de estudo privada

- **Requisito:** O sistema deve permitir que o usuario tenha acesso aos Usuarios Vinculados a uma Sala de Estudo.
- **Objetivo:** Permitir que o usuario possa acessar todos os usuarios associados a sala de estudo.
- **Descrição:** Uma tela para que o usuario proprietário tenha acesso a lista de todos os usuarios associados a sala de estudo, alem de acesso a botões para manipula-las.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario acessa a tela de Lista de Usuarios Vinculados;
 2. O aplicativo deve enviar uma requisição para API;
 3. O aplicativo deve receber a resposta da requisição e valida-la;

4. O aplicativo deve listar todos os usuarios presentes na resposta da API;
5. Caso de uso encerrado;

– **Fluxo alternativo:**

4. O aplicativo informa que não possui nenhum usuario vinculado e exibe um botão para vincular um usuario;
5. Caso de uso encerrado;

1.4.2.18 **UC0018:** Adicionar usuarios em uma sala de estudo privada

- **Requisito:** O sistema deve permitir que o usuário que criou uma sala de estudo privada adicione outros usuarios.
- **Objetivo:** Permitir que o usuario adicione outros usuarios a suas salas de estudo privada.
- **Descrição:** Uma tela onde o usuario pode informar o email de um usuario para dar acesso a sala de estudo privada.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**

– **Fluxo principal:**

1. O usuario acessa a tela de Vincular Usuario a Sala de Estudo;
2. O usuário deve informar um email valido de um usuario ja cadastrado;
3. O usuário deve enviar o formulario;
4. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
5. O aplicativo deve enviar uma requisição para API;
6. A API valida o email do usuario e vincula a sala de estudo selecionada;
7. A API retorna o resultado para a aplicação;
8. O aplicativo deve receber a resposta da requisição e valida-la;
9. O aplicativo deve informar em tela o resultado;
10. Caso de uso encerrado.

1.4.2.19 UC0019: Remover usuarios de uma sala de estudo privada

- **Requisito:** O sistema deve permitir que o usuário que criou uma sala de estudo privada remova usuarios da sala.
- **Objetivo:** Permitir que o usuario remova outros usuarios de suas salas de estudo privada.
- **Descrição:** Por meio de um botão o usuario pode desassociar outros usuarios que estão vinculados a sua sala de estudo privada.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário clica no botão de deletar sala de estudo;
 2. O aplicativo deve confirmar a intenção do usuario de deletar aquela sala de estudo;
 3. O aplicativo deve enviar uma requisição para API;
 4. A API valida e desvincula o usuario da sala de estudo selecionada;
 5. A API retorna o resultado para a aplicação;
 6. O aplicativo deve receber a resposta da requisição e valida-la;
 7. O aplicativo deve atualizar a Lista de Usuarios Vinculados;
 8. Caso de uso encerrado.
 - **Fluxo alternativo:**
 7. O aplicativo deve informar em tela ao usuario caso a resposta da API apresente algum problema ao desvincula a usuario;
 8. Caso de uso encerrado.

1.4.2.20 UC0020: Editar usuario

- **Requisito:** O sistema deve permitir que o usuário faça a edição dos seus dados de conta.
- **Objetivo:** Permitir que o usuario edite os dados de sua conta.
- **Descrição:** Uma tela que possui um formulario para que os usuarios alterem seus dados.

- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** Não tem.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuario acessa a tela de Editar de Dados;
 2. O aplicativo deve enviar uma requisição para API para buscar os dados do usuario da sessão;
 3. O aplicativo deve preencher os campos com os dados do usuario;
 4. O usuário deve informar todos os dados obrigatorios nos campos corretos do formulario;
 5. O usuário deve enviar o formulario;
 6. O aplicativo deve validar se todos os campos obrigatorios foram preenchidos;
 7. O aplicativo deve enviar uma requisição para API;
 8. A API realiza a edição do usuario e retornar o resultado para a aplicação;
 9. O aplicativo deve receber a resposta da requisição e valida-la;
 10. O aplicativo deve mostrar o resultado da edição atraves de uma mensagem em tela;
 11. Caso de uso encerrado.
- **Requisitos de tela:**

A fig. 21 apresenta a tela para edição de dados da conta do usuário logado.



Fig. 21. Tela de edição de dados do usuário logado

1.4.2.21 UC0021: Deletar usuario

- **Requisito:** O sistema deve permitir que usuario delete sua conta.
- **Objetivo:** Permitir que o usuario delete sua conta do aplicativo.
- **Descrição:** O usuario pode deletar seu usuario removendo assim todos seus dados cadastrados.
- **Ator:** Usuário.
- **Pré-Condição:** O usuario deve estar autenticado no aplicativo.
- **Pós-Condição:** O usuario deve ser redirecionado a tela de acesso do aplicativo.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário clica no botão de deletar conta;
 2. O aplicativo deve confirmar a intenção do usuario de deletar sua conta;
 3. O aplicativo deve enviar uma requisição para API;
 4. A API realiza a deleção da conta e retorna o resultado para a aplicação;
 5. O aplicativo deve receber a resposta da requisição e valida-la;

6. O aplicativo apaga o token de autenticação do usuário;
7. O aplicativo redireciona o usuário para a Tela de Acesso do aplicativo;
8. Caso de uso encerrado.

– **Fluxo alternativo:**

6. O aplicativo deve informar em tela ao usuário caso a resposta da API apresente algum problema ao deletar a conta do usuário;
7. Caso de uso encerrado.

• **Requisitos de tela:**

A fig. 22 apresenta a tela para que o usuário possa deletar sua conta e apagar seus dados do sistema.

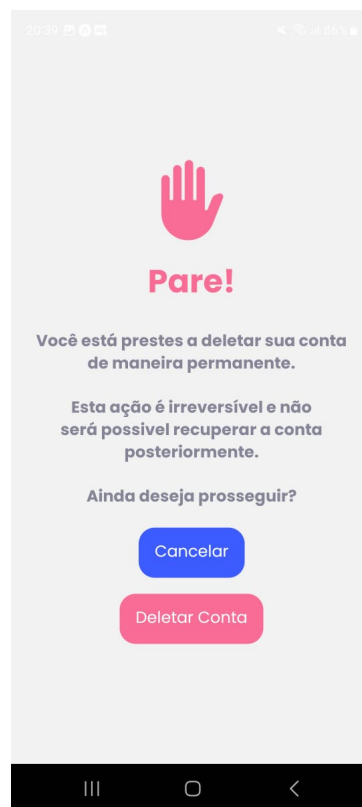


Fig. 22. Tela de apagar conta do usuário

1.4.2.22 **UC0022:** Encerrar Sessão no sistema

- **Requisito:** O sistema deve permitir que o usuário encerre sua sessão.
- **Objetivo:** Permitir ao usuário a opção de encerrar a sessão no aplicativo.
- **Descrição:** O usuário pode encerrar a sessão no aplicativo.
- **Ator:** Usuário.

- **Pré-Condição:** Estar autenticado no sistema.
- **Pós-Condição:** O usuario deve ser redirecionado a tela de acesso do aplicativo.
- **Fluxos de eventos:**
 - **Fluxo principal:**
 1. O usuário deve clicar no botão de sair;
 2. O aplicativo deve confirmar a intenção do usuario de encerrar sua sessão no aplicativo;
 3. O aplicativo deve enviar uma requisição para API;
 4. O aplicativo deve receber a resposta da requisição e valida-la;
 5. O aplicativo apaga o token de autenticação do usuario;
 6. O aplicativo redireciona o usuario para a Tela de Acesso do aplicativo;
 7. Caso de uso encerrado.
- **Requisitos de tela:**

A fig. 23 apresenta a tela onde o usuário pode encerrar sua sessão no aplicativo.

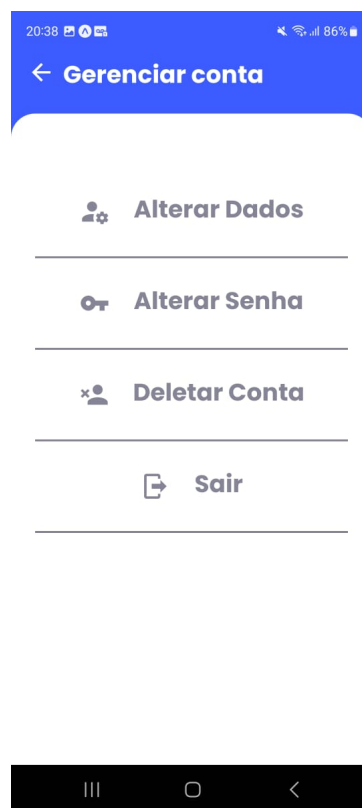


Fig. 23. Tela para encerrar sessão do usuário

1.5 Resultados e Discursão

Ao considerar a literatura revisada, foi possível constatar que alguns aplicativos móveis utilizam a técnica de repetições espaçadas de forma eficiente para promover o aprendizado ubíquo. Esses aplicativos são projetados para fornecer aos usuários uma experiência personalizada de aprendizado, adaptada às suas necessidades e objetivos específicos.

Outros estudos revisados indicam que o uso de aplicativos móveis para o aprendizado ubíquo pode levar a melhorias no desempenho dos alunos. Além disso, a revisão da literatura destacou a importância de se considerar as características do usuário e do conteúdo a ser aprendido ao desenvolver aplicativos móveis para o ensino e aprendizado.

Nosso aplicativo então se utilizou desses vários conceitos para entregar uma experiência simples e rápida ao usuário, para que se torne uma ferramenta que possa ser facilmente utilizada no dia a dia aderindo assim ao conceito do aprendizado ubíquo. O desenvolvimento também se preocupou com a acessibilidade da aplicação para torná-la mais abrangente possível.

Com relação ao objetivo proposto o aplicativo entrega uma experiência funcional porém ainda bem simplificada, devido ao tempo para desenvolvimento do trabalho não foi possível explorar muito conceitos e nem fazer um projeto com um escopo muito grande, contudo a aplicação desenvolvida apresenta um resultado satisfatório para o que se propôs.

O aplicativo no entanto ainda não é um produto, porém já tem uma base bem promissora, sendo assim ainda seria necessários testes com usuários reais para validar todas as estratégias utilizadas no desenvolvimento e traçar as melhorias e próximos passos para evoluir e transformar a aplicação em um produto completo.