CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

Faculdade de Tecnologia Baixada Santista Rubens Lara

Curso Superior de Tecnologia em
Sistemas para Internet

BARBARA HELLEN DA SILVA PEREIRA
BRUNA COSTA PINHEIRO DE SOUZA
NATHÁLIA REGINA VIEIRA TEIXEIRA

CalendAll

Gerenciador Acadêmico

Santos, SP
2024

BARBARA HELLEN DA SILVA PEREIRA
BRUNA COSTA PINHEIRO DE SOUZA
NATHÁLIA REGINA VIEIRA TEIXEIRA

CalendAll
Gerenciador Acadêmico

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia Rubens Lara, como exigência para a obtenção do Título de Tecnólogo em Sistemas para Internet.

Orientador: Prof. Felipe Cannarozzo Lourenço

Santos, SP
2024

RESUMO

Desenvolvimento de um aplicativo móvel que auxilia na organização dos alunos em suas atividades acadêmicas e facilita a interação entre eles dentro da instituição. O objetivo deste projeto é criar uma aplicação que ajude os alunos a lidarem com a procrastinação e a evitar o esquecimento de prazos de entrega. O aplicativo oferecerá recursos como a personalização de notificações para cada tarefa, a possibilidade de participar de grupos com colegas para gerenciar atividades em conjunto.

Palavras-chaves: Aplicativo móvel. Gestão acadêmica. Organização. Notificações.

ABSTRACT

Development of a mobile application that assists students in organizing their academic activities and facilitates interaction among them within the institution. The aim of this project is to create an application that helps students deal with procrastination and avoid missing deadlines. The application will offer features such as customization of notifications for each task, and the ability to participate in groups with peers to manage activities together.

Keywords: Mobile application. Academic management. Organization. Notifications.

LISTA DE ABREVIAÇÕES E SIGLAS

API	*Application Programming Interface*
APP	*Application*
CRUD	*Create, Read, Update, Delete*
DER	Diagrama de Entidade-Relacionamento
et al.	*et alii (*e outros*)*
iOS	*iPhone Operation System*
JVM	*Java Virtual Machine*
JWT	*JSON Web Token*
MER	Modelo de Entidade-Relacionamento
PNAD	Pesquisa Nacional por Amostra de Domicílios
RF	Requisitos Funcionais
RNF	Requisitos Não Funcionais
SQL	*Structured Query Language*
WAMMI	*Website Analysis and Measurement Inventory*

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

A gestão eficaz do tempo é uma necessidade premente para os estudantes universitários, sobretudo diante dos desafios inerentes à conciliação entre os estudos, o trabalho e outras obrigações pessoais. Este contexto é exacerbado pela prevalência da procrastinação, uma questão complexa que afeta significativamente o desempenho acadêmico e o bem-estar dos estudantes.

Geara et al. (2017) destacam que a procrastinação acadêmica pode ser motivada por dois grupos de fatores distintos. O primeiro, denominado Procrastinação-Desmotivação, está associado à preguiça, à percepção de dificuldade nas tarefas, ao cansaço e à falta de energia. Já o segundo grupo, Procrastinação-Ansiedade, refere-se ao medo de falhar, à dificuldade em começar tarefas e à auto expectativa elevada. Esses aspectos têm impactos diretos sobre o desempenho dos alunos, conforme evidenciado por estudos como os de Ribeiro et al. (2014) e Silva et al. (2016), que apontam para notas mais baixas entre os procrastinadores, além de uma autoavaliação negativa do desempenho acadêmico (Amaro et al., 2016; *Semprebon* et al., 2017).

A necessidade de conciliar estudos e trabalho é uma realidade para muitos jovens universitários, como indicado pelos dados do Boletim da Educação – PNAD Contínua (2023), que revelam que 11,8% dos jovens de 15 a 29 anos no Brasil estão nessa situação. Essa dupla jornada pode acarretar sobrecarga e estresse, dificultando ainda mais a administração do tempo e potencializando os efeitos da procrastinação.

Diante desses desafios, surge a importância de ferramentas eficazes de gerenciamento acadêmico, como o objetivo de otimizar a gestão do tempo dos estudantes, concentrando todas as datas importantes da vida acadêmica em um único local acessível, buscando facilitar a organização coletiva por meio da criação de grupos, garantindo uma comunicação eficiente e colaboração entre os membros da turma.

O propósito de criar um aplicativo de gestão acadêmica é proporcionar uma experiência abrangente e integrada de gerenciamento acadêmico para estudantes universitários. Que vise não apenas otimizar a gestão do tempo, mas também promover a colaboração e comunicação entre os membros da turma. Que busque

fomentar a socialização entre os alunos, facilitando a organização na hora de suas entregas.

Com base nos tópicos apresentados, surgiu a ideia do CalendAll, uma plataforma projetada para auxiliar os alunos de ensino superior na organização e no enfrentamento dos desafios relacionados à procrastinação e ao esquecimento das tarefas acadêmicas. O aplicativo oferece uma variedade de ferramentas e recursos abrangentes para lidar com as demandas da vida universitária, especialmente no que diz respeito aos prazos de entrega.

A plataforma foi desenvolvida com foco na intuitividade e na facilidade de uso, tornando-a acessível para todos os alunos das instituições de ensino. Uma de suas características distintivas é a capacidade de personalização das notificações via *push*, permitindo que os alunos definam suas prioridades e recebam lembretes de acordo com suas preferências individuais. Através do aplicativo, os usuários podem acompanhar facilmente prazos de entrega, eventos e tarefas acadêmicas, com a possibilidade de personalizar o estilo e a frequência de notificações, além de definir prioridades distintas.

Além disso, o CalendAll facilita a ampliação da rede de contatos dos alunos dentro de suas universidades, proporcionando uma troca eficaz de conhecimento e experiências entre os estudantes. Isso promove um ambiente de colaboração e apoio mútuo, contribuindo para o sucesso acadêmico e pessoal dos usuários. Através do aplicativo, é oferecida a troca de conhecimentos por meio da criação de eventos públicos acessíveis a todos os integrantes da sala, garantindo um ambiente colaborativo que enriquece a experiência acadêmica.

Em resumo, o CalendAll é uma ferramenta essencial para os alunos universitários enfrentarem os desafios da vida acadêmica universitária, garantindo uma gestão eficiente do tempo, combatendo à procrastinação e estimulando à colaboração e ao compartilhamento de conhecimento.

1.1 OBJETIVO

Nessa seção, será fornecida uma visão abrangente dos objetivos gerais e específicos do projeto, detalhando as métricas utilizadas para sua concretização

1.1.1 OBJETIVO GERAL

O objetivo geral do aplicativo CalendAll é aprimorar a organização acadêmica dos estudantes universitários, oferecendo ferramentas de suporte para o gerenciamento eficaz de suas rotinas de estudo. Através do aplicativo, os usuários podem acompanhar facilmente prazos de entrega, eventos e tarefas acadêmicas, com a possibilidade de personalizar o estilo e a frequência de notificações, além de definir prioridades distintas, que apresentam estilos visuais distintos para facilitar a diferenciação.

Adicionalmente, o CalendAll busca incentivar o networking universitário, facilitando interações entre alunos da mesma turma e promovendo a troca de conhecimentos por meio da criação de eventos públicos acessíveis a todos os integrantes da sala. Dessa forma, o aplicativo cria um ambiente colaborativo que enriquece a experiência acadêmica, proporcionando aprendizado além da sala de aula e contribuindo para o desenvolvimento de habilidades valiosas para as carreiras futuras dos estudantes.

1.1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos do projeto incluem:

- Identificação das necessidades de alunos e representantes de turma de entidades de Ensino Superior, através de pesquisas e análises quanto a organização acadêmica universitária;
- Estudo sobre tecnologias e aplicações semelhantes ao objetivo do CalendAll, utilizando os sistemas do *Microsoft Teams*, *Google Classroom* e *Moodle* como objetivos de pesquisa;
- Desenvolvimento de aplicação de gerenciamento acadêmico, com foco em auxiliar estudantes na gestão de sua jornada universitária.

1.2 ESTADO DA ARTE

O aplicativo é baseado em funcionalidades encontradas em outras aplicações com objetivos semelhantes. O *Microsoft Teams*, ilustração 1, de acordo com a

documentação oficial, é uma ferramenta extremamente útil no contexto educacional para gerenciar e atribuir tarefas aos alunos. Na plataforma, os professores podem criar canais específicos para cada disciplina ou turma, onde podem compartilhar materiais de estudo, enviar anúncios e atribuir tarefas aos estudantes.

Uma das funcionalidades mais relevantes para atribuição de tarefas é a aba "Atribuições" dentro de cada canal. Nesse espaço, os professores podem criar tarefas, definir prazos de entrega, adicionar instruções detalhadas e até mesmo anexar arquivos relevantes. Os alunos, por sua vez, recebem notificações sobre as novas tarefas atribuídas e podem acessá-las facilmente a partir do *Teams*.

Ilustração 1 – Microsoft Teams (logo)

Fonte: https://teams.microsoft.com/, 2024

O *Google Classroom*, ilustração 2, de acordo com a documentação oficial, é uma plataforma de ensino desenvolvida pelo *Google*, é projetado para simplificar a interação entre professores e alunos, bem como para atribuir, entregar e avaliar tarefas escolares de forma eficiente. Similar ao Microsoft Teams, o sistema oferece um ambiente virtual onde os professores podem criar salas de aula virtuais para cada uma de suas turmas. Dentro dessas salas de aula, eles podem compartilhar materiais de estudo, postar links úteis, fazer anúncios e atribuir tarefas aos alunos.

Ilustração 2 – Google Classroom (logo)

Fonte: https://classroom.google.com/, 2024

A funcionalidade central é a habilidade de criar e gerenciar tarefas. Os professores podem criar tarefas, estabelecer prazos de entrega, adicionar descrições detalhadas e anexar arquivos relevantes, como documentos, apresentações ou vídeos. Os alunos, por sua vez, recebem notificações sobre as novas tarefas e podem acessá-las diretamente da aplicação.

Moodle, ilustração 3, de acordo com a documentação oficial, é uma plataforma parecida com as citadas anteriormente. É amplamente usada por instituições de ensino e assim como o *Microsoft Teams* e o *Google Classroom* possui funcionalidades para a atribuição e gerenciamento de tarefas. Os professores criam as atividades e dão prazos de entrega para cada uma das criadas, podem adicionar descrições detalhadas e anexar materiais de apoio, como arquivos, links e vídeos. Também é possível realizar a configuração de tarefas de diferentes tipos como fóruns de discussão, tarefas práticas, envio de documentos e até *quizzes.*

Ilustração 3 - Moodle (logo)

Fonte: https://moodle.org/

As análises das funcionalidades dos sistemas mencionados foram essenciais para o desenvolvimento do CalendAll. Inspirando-se nessas plataformas, incorporaram-se recursos de gestão de tarefas, criando uma solução prática e eficiente para atender às necessidades dos usuários em sua organização acadêmica.

2. DESENVOLVIMENTO

Nesse capítulo será abordado as fases do desenvolvimento do sistema proposto. Para tanto, esse capítulo será subdividido nos seguintes itens; análise do sistema, a estrutura do Banco de Dados, as ferramentas utilizadas para o desenvolvimento do *front-end* e para o *back-end***.**

2.1 ANÁLISE DO SISTEMA

As linguagens utilizadas no desenvolvimento do aplicativo serão:

React Native, ilustração 4, é um *framework* de desenvolvimento de aplicativos móveis que possibilita a criação de aplicativos nativos para *iOS (Iphone Operational System)* e *Android* utilizando *JavaScript* e *React*. Ele permite que os desenvolvedores construam interfaces de usuário robustas e fluidas, aproveitando uma abordagem de desenvolvimento baseada em componentes reutilizáveis.

De acordo com William Danielsson (2016), *o* React Native se diferencia por permitir que o código JavaScript interaja com componentes nativos, proporcionando uma experiência de usuário fluida. Ele suporta a execução assíncrona e multitarefa, permitindo que operações sejam realizadas em segundo plano sem comprometer o desempenho da interface.

Essa abordagem, junto com o suporte a APIs de plataforma e a possibilidade de criar módulos nativos personalizados, oferece flexibilidade e consistência no design e na experiência de uso.

Ilustração 4 **–** *React Native* (logo)

Fonte: https://reactnative.dev/

TypeScript, ilustração 5, segundo a documentação oficial, é uma extensão de *JavaScript* desenvolvida pela *Microsoft*, adicionando tipos estáticos opcionais à linguagem. É útil para desenvolver aplicativos *JavaScript* em grande escala, oferecendo detecção de erros em tempo de compilação e aumentando a produtividade. Com *TypeScript*, os desenvolvedores podem declarar tipos para variáveis e outros elementos do código, proporcionando segurança e clareza adicionais.

Ilustração 5 – *TypeScript* (logo)

Fonte: https://www.typescriptlang.org/

Spring Boot, ilustração 6, é um *framework* do *Spring* que facilita a criação de aplicativos Java autônomos, prontos para produção, com o mínimo de configuração possível. Segundo afirmado na documentação oficial, alguns de seus objetivos consistem em fornecer uma experiência mais rápida e acessível de início para todo o desenvolvimento *Spring*, além de oferecer uma gama de funções não-funcionais que são comuns a grandes classes de projetos.

Ilustração 6 – *Spring Framework* (logo)

Fonte: https://spring.io/

Com o *Spring Boot*, os desenvolvedores podem iniciar rapidamente novos projetos com uma estrutura sólida e consistentemente configurada, permitindo que se concentrem mais na lógica de negócios do que na configuração do ambiente. Isso o

torna uma escolha popular para o desenvolvimento de aplicativos *Java*, tanto para aplicativos *web* quanto para serviços de *backend*.

Java, ilustração 7, é uma linguagem de programação amplamente utilizada, conhecida por sua portabilidade, robustez e segurança. Ela é usada em uma variedade de aplicações, desde desenvolvimento de *software* empresarial até aplicações móveis e embarcadas.

> Java é a linguagem de programação e plataforma de desenvolvimento mais prolífica do mundo. Reduz custos e prazos de desenvolvimento, impulsiona a inovação e aprimora os serviços de aplicações. Com milhões de desenvolvedores executando mais de 60 bilhões de Java *Virtual Machines* em todo o mundo, o Java continua sendo a plataforma de desenvolvimento preferida de empresas e desenvolvedores. (ORACLE, [s.d.])

Java é uma linguagem orientada a objetos, o que significa que ela organiza o código em objetos que interagem entre si. Além disso, uma das características distintivas do Java é a sua máquina virtual Java (JVM – Java *Virtual Machine*), que permite que o código Java seja executado em qualquer plataforma que tenha uma JVM instalada, tornando-o altamente portátil. Java possui uma vasta biblioteca padrão, oferecendo suporte para uma variedade de funcionalidades.

Ilustração 7 – Java (logo)

Fonte: https://www.oracle.com/java/

MySQL, ilustração 8, abreviação de *Structured Query Language*, é uma linguagem de programação utilizada para gerenciar bancos de dados relacionais. Essa escolha se baseia nos diversos recursos que oferece para manipulação de dados, incluindo operações de armazenamento, atualização, remoção e recuperação. Além disso, o *MySQL* é conhecido por fornecer segurança avançada, escalabilidade

e desempenho, o que o torna uma opção ideal para lidar com os dados coletados pela aplicação.

Ilustração 8 – MySQL (logo)

Fonte: www.mysql.com

2.1.1 ANÁLISE DE REQUISITOS

A análise de requisitos é uma fase essencial no desenvolvimento de *software*. Para tanto, geralmente são dois quadros. Um deles, refere-se aos RNF (Requisitos Não Funcionais) do sistema que abordam tópicos que não estão relacionados diretamente as funcionalidades do sistema, mas sim como deve se comportar ou a qualidade que deve apresentar, conforme a ilustração 9.

Ilustração 9 – Quadro de Requisitos Não Funcionais

[RNF01]	O aplicativo deve carregar a tela inicial em até 4 segundos em conexões de internet padrão
[RNF02]	As notificações push devem ser entregues aos usuários em no máximo 12 segundos após o envio.
[RNF03]	Políticas de privacidade devem ser claramente definidas e comunicadas aos usuário
[RNF04]	O aplicativo deve ser compatível com dispositivos móveis Android (versão 8.0 e superiores) e iOS (versão 11.0 e superiores).

Fonte: Elaborado pelos autores, 2024

O outro quadro, refere-se aos RF (Requisitos Funcionais), estes são responsáveis por descrever as características, comportamento e funcionalidade que o software deve ter para atender as necessidades do usuário. São a base do que a aplicação deve realizar, conforme mostrado na ilustração 10.

Ilustração 10 – Quadro de Requisitos Funcionais

[RF01]	O sistema deve permitir que os usuários se cadastrem utilizando um endereço de e-mail válido, uma senha, um nome e uma data de nascimento válida.
[RF02]	O sistema deve permitir que os usuários façam login utilizando o endereço de e-mail e senha cadastrados.
[RF03]	O sistema deve permitir que os usuários adicionem novas tarefas/eventos com informações como título, descrição, data, hora, prioridade e tipo de notificação.
[RF04]	O sistema deve permitir que os usuários editem e excluam tarefas/eventos previamente adicionados.
[RF05]	O sistema deve permitir que os usuários personalizem as notificações para diferentes tipos de tarefas e eventos.
[RF06]	O sistema deve enviar notificações push para lembrar os usuários sobre as tarefas próximas do prazo.
[RF07]	O sistema deve oferecer uma visualização de calendário onde os usuários possam ver todas as suas tarefas e eventos agendados.
[RF08]	O sistema deve permitir que os usuários criem e gerenciem salas com outros usuários.
[RF09]	O sistema deve permitir que apenas os usuários com permissão de administrador criem tarefas/eventos para as salas.
[RF10]	O sistema deve permitir que os usuários definam suas preferências de privacidade e notificações.

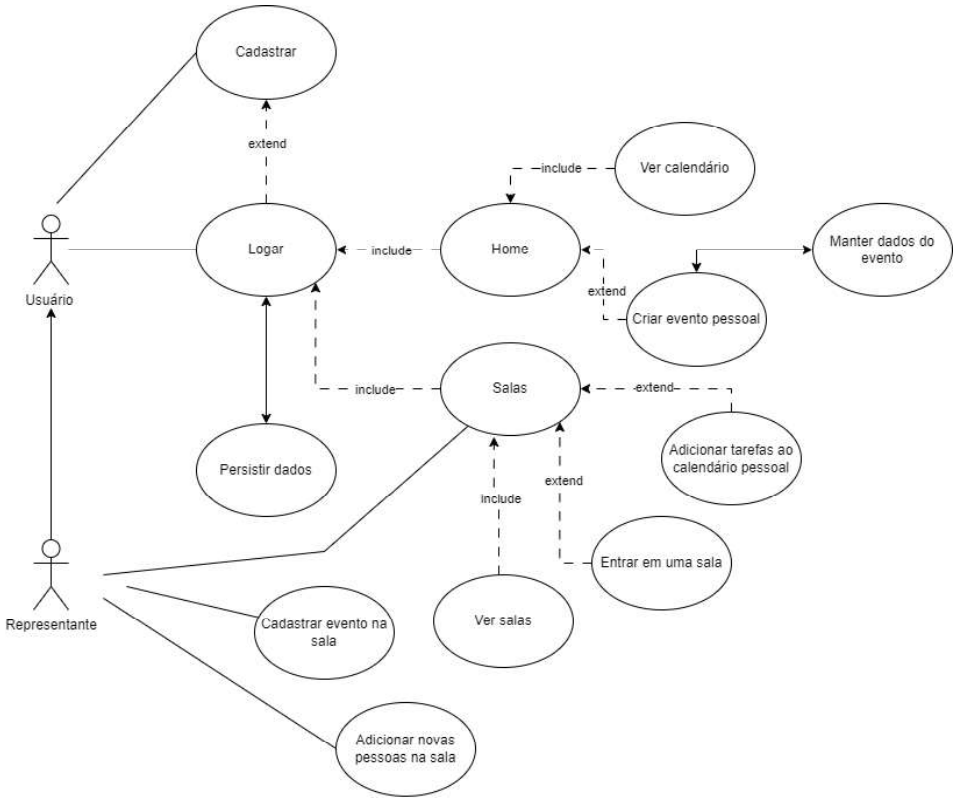
Fonte: Elaborado pelos autores, 2024

2.1.2 DIAGRAMA DE CASO DE USO

Um diagrama de caso de uso, ilustração 11, é uma representação visual utilizada na engenharia de *software* para ilustrar as interações entre os usuários (atores) e um sistema, descrevendo as funcionalidades principais do sistema do ponto de vista do usuário final. Este diagrama ajuda a identificar e organizar os requisitos funcionais do sistema, destacando os diferentes casos de uso, que são cenários específicos nos quais os usuários interagem com o sistema para atingir um objetivo particular. Ele facilita a comunicação entre desenvolvedores, clientes e outros *stakeholders*, garantindo uma compreensão clara e compartilhada das funcionalidades que o sistema deve oferecer.

Além disso, o diagrama de caso de uso é fundamental para a análise de requisitos do sistema, pois permite que os integrantes do projeto tenham uma rápida visualização das diferentes operações do sistema em cada situação.

Ilustração 11 – Diagrama de Caso de Uso



Fonte: Elaborado pelos autores, 2024

2.1.3 FLUXO DE EVENTOS

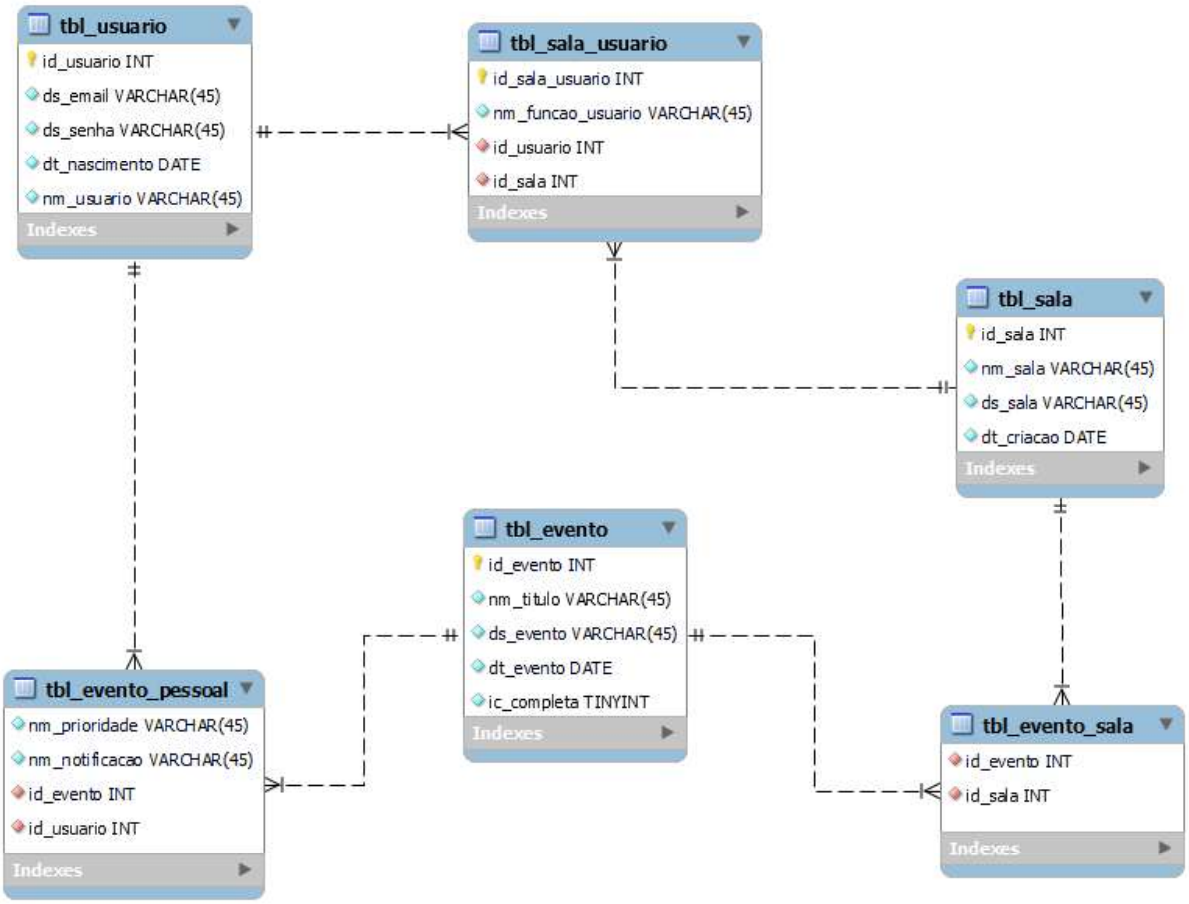
O fluxo de eventos tem por objetivo caracterizar o início e o término do caso de uso, estabelecendo a interação dos atores no processo descrito. Dessa forma, é gerado um fluxo principal e um fluxo alternativo para o comportamento de cada caso de uso do sistema. A descrição detalhada dos fluxos de eventos deste sistema pode ser consultada no Apêndice A.

2.2 BANCO DE DADOS

O modelo entidade-relacionamento (MER), conforme ilustração 12, é uma técnica de modelagem de dados amplamente utilizada para representar visualmente a estrutura e as interações dentro de um banco de dados. Este modelo auxilia na definição detalhada das entidades, seus atributos específicos e os relacionamentos

que existem entre elas. Ele é essencial para projetar a arquitetura de um banco de dados, permitindo uma comunicação clara e precisa sobre a estrutura e a dinâmica dos dados entre desenvolvedores, analistas e *stakeholders*. Além disso, o MER facilita a identificação de redundâncias e inconsistências, garantindo um *design* de banco de dados eficiente e bem estruturado.

Ilustração 12 – MER (Modelo Entidade-Relacionamento)

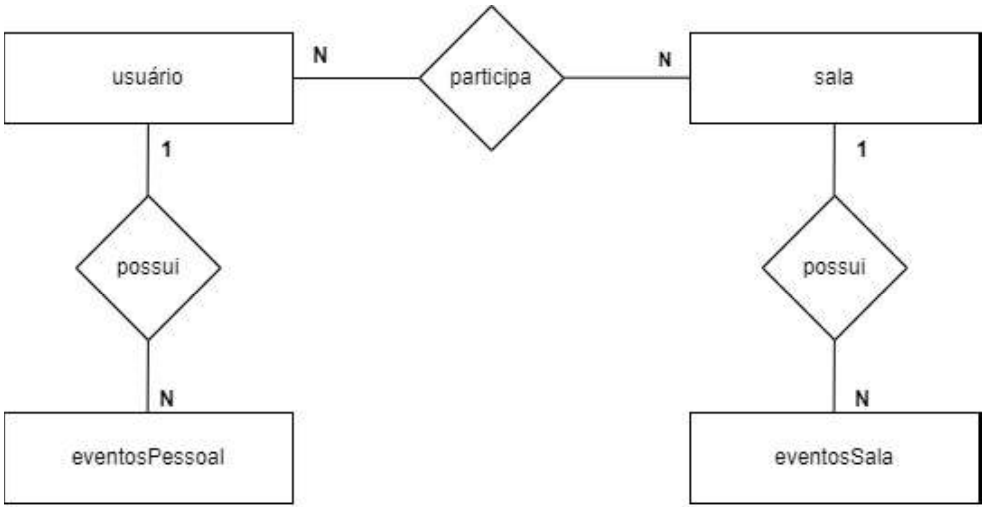


Fonte: Elaborado pelos autores, 2024

DER, ou Diagrama de Entidade-Relacionamento, ilustração 13, é uma representação visual fundamental usada para modelar dados em sistemas de banco de dados. Ele mostra as entidades (objetos ou conceitos) e os relacionamentos entre elas, incluindo suas propriedades e associações. Além de facilitar a compreensão da estrutura e das interações dentro do banco de dados, o DER também serve como uma ferramenta de comunicação entre analistas, desenvolvedores e partes interessadas, permitindo que todos tenham uma visão clara do sistema. Essa representação é

crucial durante a fase de design, pois ajuda a identificar redundâncias e inconsistências nos dados, promovendo um design mais eficiente e organizado.

Ilustração 13 – DER (Diagrama de Entidade-Relacionamento)



Fonte: Elaborado pelos autores, 2024

2.3 CAMADA DE NEGÓCIO

A principal funcionalidade do CalendAll é o gerenciamento centralizado de tarefas e eventos acadêmicos, permitindo que os usuários personalizem prioridades e notificações conforme suas necessidades. Para viabilizar essa operação, foi implementada uma API *(Application Programming Interface)*, desenvolvida em Java com o uso do *Spring Boot*, que estabelece a comunicação entre o *front-end* e o banco de dados. Essa API é responsável por gerenciar as principais entidades do sistema, como Usuário, Sala e Evento, oferecendo uma estrutura robusta para a interação entre os componentes da aplicação.

> API serve para interligar, fornecer suas funcionalidades ou rotinas, para mais de um sistema, sem que as partes saibam detalhes interno de sua implementação. Um grande exemplo é o serviço de geolocalização do *Google Maps*, que muitos sistemas (clientes) utilizam seu serviço, como aplicativos de *delivery*, transporte, trânsito e outros vários, e nenhum deles sabe detalhes de sua implementação. (JUNIOR, ROCHA, MACIEL, 2020, p. 501)

No contexto do CalendAll, os usuários podem se inscrever em salas virtuais onde, juntamente com outros participantes, podem criar e compartilhar eventos públicos. Esses eventos podem ser adicionados aos calendários pessoais dos

usuários, promovendo uma experiência de maior personalização. Dentro dessas salas, os usuários assumem diferentes papéis:

- **Representante da sala**: é o administrador principal e responsável pela criação e gestão da sala e de seus eventos;
- **Vice-representante da sala:** indicado pelo representante, é o segundo administrador, auxiliando na gestão do ambiente;
- **Aluno comum:** participa da sala como membro, podendo visualizar os eventos, mas sem permissões de edição ou gestão.

Além das salas, o CalendAll oferece uma área dedicada ao calendário pessoal de cada usuário, com total possibilidade de personalização. Nessa interface, o usuário pode criar eventos privados e configurar notificações de acordo com suas preferências, determinando a prioridade e a frequência dos alertas. Os eventos das salas nas quais o usuário participa também podem ser incorporados ao calendário pessoal, desde que o próprio usuário opte por isso.

A seguir, serão apresentados os detalhes do desenvolvimento da API que sustenta o CalendAll, garantindo a funcionalidade de todos os recursos mencionados acima.

2.3.1 ARQUITETURA DA CAMADA DE NEGÓCIO

Cada camada da API desempenha um papel específico, garantindo a organização, manutenibilidade e segurança do sistema. A *Controller* é a camada responsável por processar as requisições feitas pelos usuários na aplicação, enquanto a camada *Service* centraliza toda a lógica de negócio.

Na *Controller* de autenticação, como representado na ilustração 14, os *endpoints* de *login* e cadastro são gerenciados, além da opção de redefinir a senha do usuário. Por meio da injeção de dependência da camada *Service,* as validações dos dados fornecidos pelo usuário são realizadas, incluindo a aplicação de *hash* na senha e a geração de um *token* de autenticação no formato JWT (*JSON Web Token*), utilizado para autorizar e identificar o usuário em futuras requisições, permitindo o acesso seguro ao sistema.

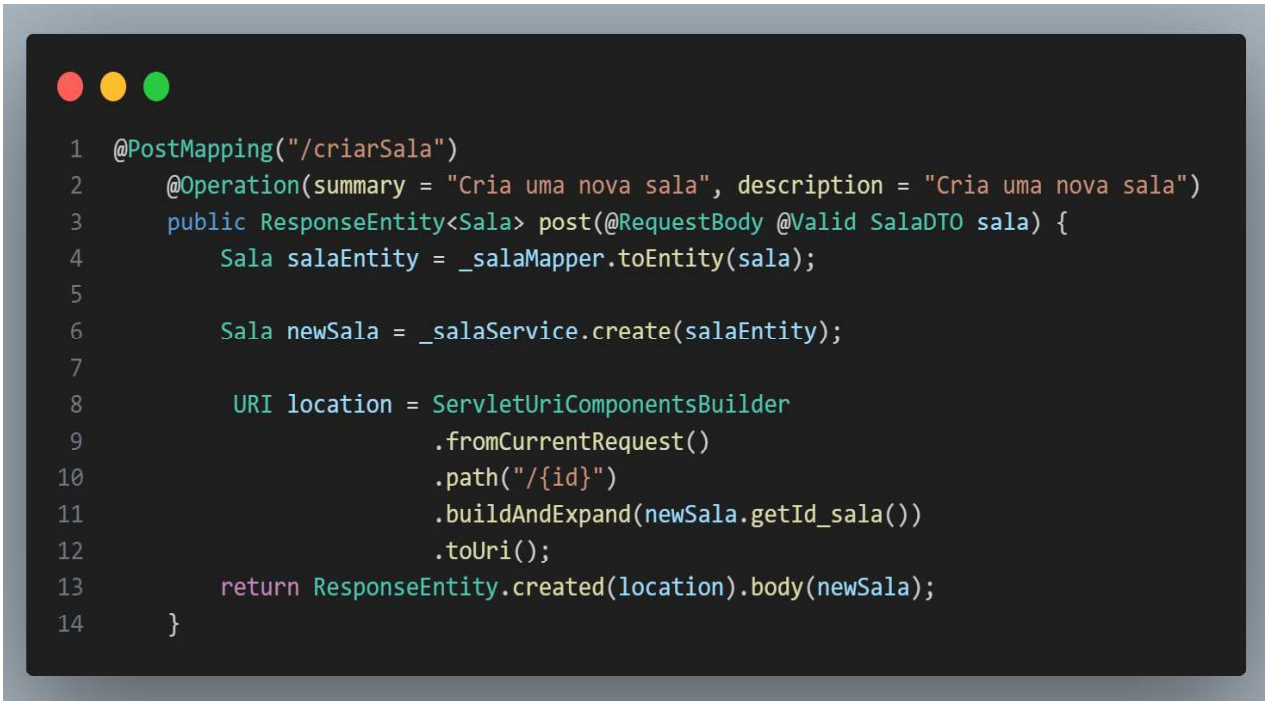
Ilustração 14 – Trecho do código de autenticação do usuário

```
public class AutenticacaoController {

    private final UsuarioService _usuarioService;
    private final TokenService _tokenService;

    @PostMapping("/login")
    public ResponseEntity<?> login(@RequestBody LoginDto loginDto) {

        try {
            var usuario = _usuarioService.AutenticarUsuario(loginDto);

            String token = _tokenService.GerarToken(usuario);

            return ResponseEntity.ok(new TokenJwtDto(token));

        } catch (Exception ex) {

            return ResponseEntity.badRequest().body(ex.getMessage());
        }
    }

    @PostMapping("/cadastro")
    public ResponseEntity<?> cadastrar(@RequestBody CadastroDto cadastroDto) {

        try {
            Usuario novoUsuario = _usuarioService.CadastrarUsuario(cadastroDto);
            String token = _tokenService.GerarToken(novoUsuario);

            return ResponseEntity.ok(new TokenJwtDto(token));

        } catch (Exception ex) {

            return ResponseEntity.badRequest().body(ex.getMessage());
        }
    }
}
```

Fonte: Elaborado pelos autores, 2024.

Já na *Controller* da entidade Sala, estão definidos os *endpoints* que implementam as funções CRUD (*Create, Read, Update* e *Delete*), ou seja, o código responsável pela criação, leitura, atualização e remoção das salas no sistema. Conforme demonstrado na ilustração 15, a criação de uma nova sala é realizada a partir dos atributos enviados no *body* da requisição, contendo o nome e a descrição da sala. Após receber esses dados, a *Controller* invoca a função de criação correspondente na camada *Service*, que é responsável por gerenciar a lógica de negócios e persistir a nova sala no banco de dados, gerando automaticamente o código de identificação da sala, sua data de criação, quantidade de eventos e quantidade de usuários.

Ilustração 15 – Trecho do código de criação de sala



```java
@PostMapping("/criarSala")
    @Operation(summary = "Cria uma nova sala", description = "Cria uma nova sala")
    public ResponseEntity<Sala> post(@RequestBody @Valid SalaDTO sala) {
        Sala salaEntity = _salaMapper.toEntity(sala);

        Sala newSala = _salaService.create(salaEntity);

         URI location = ServletUriComponentsBuilder
                        .fromCurrentRequest()
                        .path("/{id}")
                        .buildAndExpand(newSala.getId_sala())
                        .toUri();
        return ResponseEntity.created(location).body(newSala);
    }
```

Fonte: Elaborado pelos autores, 2024.

Os demais métodos CRUD de uma sala seguem a mesma estrutura, sendo gerenciados pela *Controller* e pela camada *Service*.

Ao criar uma sala, uma relação entre Sala e Usuário é estabelecida. O usuário responsável pela criação da sala assume automaticamente a função de Representante de Sala, com permissões de administrador sobre o ambiente. Esse representante tem a opção de adicionar outros usuários como Alunos Comuns, criando relações entre a sala e usuários sem direitos de gerenciamento.

A criação de um novo evento público dentro de uma sala, ação permitida apenas para o Representante e o Vice-Representante, segue uma lógica semelhante à criação de salas, conforme ilustração 16. A *Controller* recebe os atributos enviados no body da requisição, contendo as informações principais do evento, como título, descrição e data. Em seguida, ela invoca a função de criação correspondente na camada *Service*. A principal diferença está nas validações adicionais realizadas pela Controller, como a verificação da existência da sala fornecida, a validação da data de conclusão do evento e da permissão de administrador do usuário, garantindo que as informações estejam corretas e que o evento seja criado em conformidade com as regras de negócio.

Ilustração 16 – Trecho do código de criação de evento públicos

```java
@PostMapping("/{id_sala}/criarEvento")
    @Operation(summary = "Cria um novo evento", description = "Cria um novo evento em uma sala")
    public ResponseEntity<EventoSala> postEventoSala(@PathVariable Long id_sala, @RequestBody @Valid EventoSalaDTO eventoSala) {

        if(
            eventoSala.getDt_evento().isBefore(LocalDate.now()) ||
            eventoSala.getTitulo() == null
            ){
            return ResponseEntity.badRequest().build();
        }

        EventoSala eventoSalaEntity = _EventoSalaMapper.toEntity(eventoSala, id_sala);

        EventoSala newEventoSala = _salaService.createEventoSala(id_sala, eventoSalaEntity);

        if(newEventoSala == null){
            return ResponseEntity.notFound().build();
        }

        URI location = ServletUriComponentsBuilder
                        .fromCurrentRequest()
                        .path("/{id}")
                        .buildAndExpand(newEventoSala.getId_evento())
                        .toUri();
        return ResponseEntity.created(location).body(newEventoSala);
    }
```

Fonte: Elaborado pelos Autores, 2024.

Além dos *endpoints* apresentados, a camada de negócios do CalendAll também é responsável pelo gerenciamento dos eventos pessoais de cada usuário, proporcionando alto grau de personalização, em relação às notificações e prioridades. Esses eventos podem ser criados diretamente na tela inicial do sistema ou transferidos a partir de eventos públicos das salas em que o usuário participa, independentemente de sua função dentro dessas salas. Ao realizar essa transferência, o evento pessoal herda as propriedades do evento público original, garantindo consistência nas informações e mantendo o alinhamento entre o calendário pessoal do usuário e o calendário coletivo da sala.

2.4 CAMADA DE APRESENTAÇÃO

Neste tópico, serão apresentadas as etapas de desenvolvimento das telas do aplicativo. O design foi criado no *Figma*, uma ferramenta amplamente utilizada para o desenvolvimento de interfaces e protótipos. A implementação do *front-end*, responsável por tornar o design interativo, foi realizada utilizando o *React Native*, um *framework JavaScript* popular para a criação de aplicativos móveis. Uma das grandes vantagens do *React Native* é a possibilidade de desenvolver para sistemas como iOS

e Android com um único código-base. Além disso, foi escolhido o *TypeScript* ao invés do *JavaScript*, devido à sua capacidade de tipagem estática, o que aumenta a segurança e a confiabilidade do código. A linguagem oferece uma camada adicional de verificação de erros, reduzindo a ocorrência de bugs durante o desenvolvimento e facilitando a manutenção do código em longo prazo.

Ao abrir o aplicativo, a tela inicial, será a de login, conforme a ilustração 17, para que o usuário possa colocar as credenciais e acessar a conta na aplicação. Caso não tenha a conta, poderá se cadastrar para ter acesso.

Ilustração 17 – Tela de *login*

Fonte: Elaborado pelos autores, 2024

Após realizar o login, o usuário será encaminhado para a tela *Home*, conforme ilustração 18, no qual poderá ver o calendário e os próximos eventos.

Ilustração 18 – Tela Home do aplicativo

Fonte: Elaborado pelos autores, 2024

Ao clicar no botão criar evento, um *modal* será aberto e o usuário deverá preencher as informações necessárias para cadastro de um evento. Caso queira,

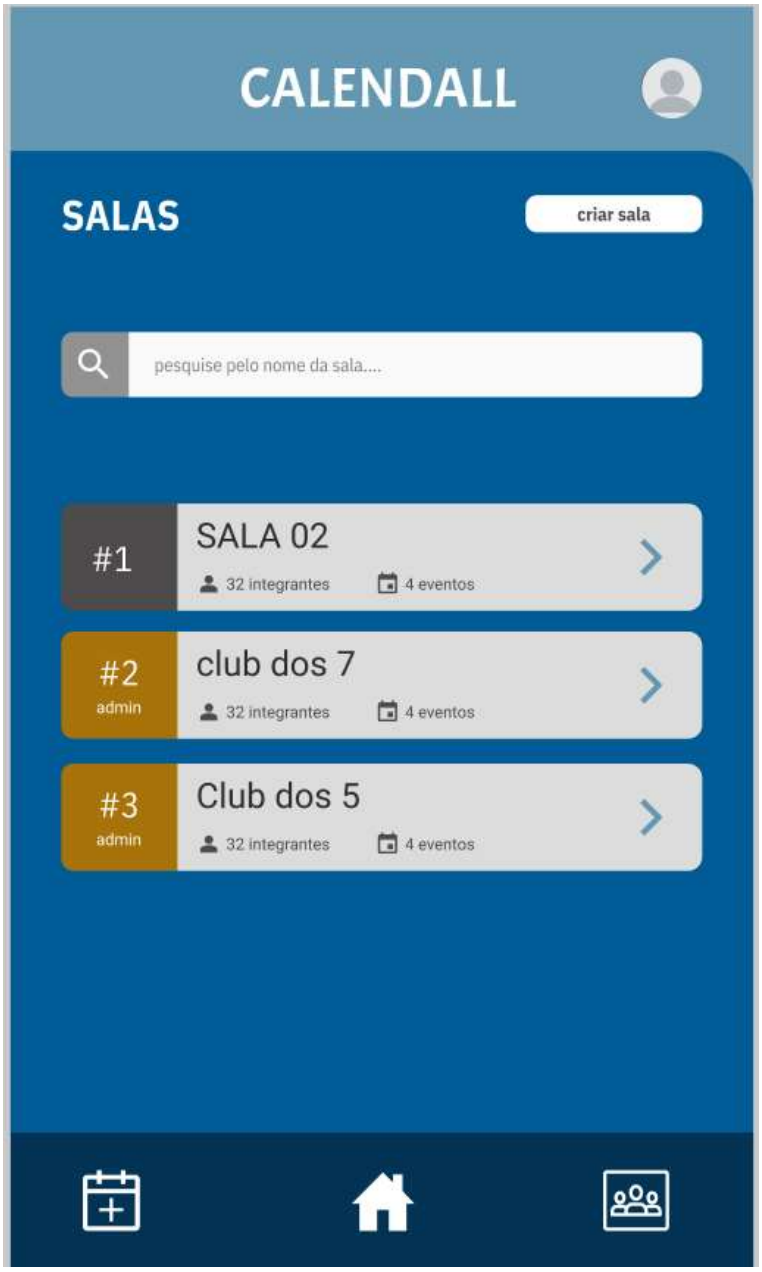
pode personalizar a quantidade de vezes que será notificado referente ao evento cadastrado, conforme a ilustração 19.

Ilustração 19 – Modal de cadastro de evento

Fonte: Elaborado pelos autores, 2024

O sistema também oferece uma funcionalidade de grupos de usuários, representando as salas. Cada sala possui dois administradores: o criador do grupo e outro usuário designado pelo criador com privilégios administrativos. O usuário pode visualizar as salas em que está participando ao clicar no ícone correspondente no menu. Se for administrador de alguma sala, a lateral esquerda da sala será destacada em âmbar escuro, com a palavra "admin" exibida abaixo do índice da sala, conforme a ilustração 20.

Ilustração 20 – Salas



Fonte: Elaborado pelos autores, 2024

Ao entrar na sala, o usuário comum poderá visualizar todos os eventos criados pelo administrador, conforme mostra a ilustração 21. Caso deseje, poderá adicionar esses eventos ao calendário pessoal e personalizar as notificações. Sempre que um novo evento for criado, o usuário será notificado e terá a opção de integrá-lo ao calendário.

Ilustração 21 – Visão da sala como usuário comum

Fonte: Elaborado pelos autores, 2024

O administrador da sala tem diversas responsabilidades e permissões. Ele pode editar os dados da sala e adicionar novos eventos. Quando um novo evento é criado, uma notificação é enviada aos usuários apenas uma vez para informá-los sobre o evento. Além disso, o administrador tem a capacidade de adicionar ou remover integrantes da sala, podendo realizar essas ações através da pesquisa pelo e-mail dos usuários ou enviando um *link* de convite para que eles se juntem à sala. Também é possível editar ou excluir eventos existentes conforme necessário. A seguir, na ilustração 22, é apresentada a visão da sala como administrador.

Ilustração 22 – Visão da sala como administrador

Fonte: Elaborado pelos autores, 2024

3. RESULTADOS

Este capítulo apresenta a conclusão do desenvolvimento da aplicação na prática, destacando a realização de testes com usuários que fazem parte do público-alvo. O principal objetivo desses testes é o de coletar *feedbacks* diretos do usuário final, essenciais para o aprimoramento da solução.

A técnica de avaliação utilizada foi o teste de usabilidade, que consistiu na criação de cenários e tarefas específicas para que os participantes interagissem com o sistema. Durante esse processo, foi feita a observação de seus comportamentos. A importância deste teste está em sua capacidade de identificar as necessidades, expectativas e dificuldades dos usuários, possibilitando o aprimoramento da experiência oferecida.

O teste de usabilidade com usuários reais é o teste mais importante e indispensável, pois permite identificar quais seus problemas exatos com a interface que está sendo testada (Nielsen, 1993). Seguindo essa abordagem, os desenvolvedores podem entender melhor quais funcionalidades são intuitivas e onde ocorrem dificuldades por parte dos usuários, permitindo a idealização de possíveis melhorias no design ou lógica de negócio da aplicação.

Assim, se tornam essenciais no processo de criação, pois não apenas revelam falhas, mas também orientam melhorias que aumentam a satisfação e a eficiência do usuário.

3.1 TESTES

O aplicativo CalendAll, junto com instruções e um questionário de avaliação, foi disponibilizado a um grupo de cinco participantes, para que pudessem avaliar as suas experiências de uso no sistema.

Durante o teste, cada participante foi solicitado a executar dez tarefas específicas. Essas tarefas foram cuidadosamente selecionadas, pois representam as principais funcionalidades do sistema. A lista completa das tarefas está apresentada na Tabela 1, a seguir.

Tabela 1 – Tarefas do Teste de Usabilidade

Nº	Tarefa
01	Realizar cadastro de usuário
02	Realizar *login* do usuário cadastrado
03	Criar nova sala
04	Adicionar novo aluno na sala criada
05	Criar um evento público na sala criada
06	Associar evento público criado ao calendário pessoal, personalizando-o
07	Criar um evento personalizado no calendário pessoal
08	Visualizar os eventos do dia no calendário pessoal
09	Visualizar todos os eventos presentes no calendário pessoal
10	Abrir tela de perfil do usuário logado

Fonte: Autores, 2024

Os usuários selecionados, estudantes universitários na faixa etária de 20 a 25 anos, de ambos os gêneros, exibiram comportamentos semelhantes ao realizar as tarefas solicitadas durante o teste. O tempo médio total para a conclusão das atividades foi de 6 minutos, evidenciando um rápido entendimento da aplicação e uma resposta ágil do sistema. Todas as tarefas foram executadas com sucesso, sem erros que impedissem sua execução.

O teste ocorreu na Baixada Santista, no estado de São Paulo, no segundo semestre do ano de 2025.

É importante destacar que todos os participantes possuíam conhecimento e experiência prévia com tecnologia, o que facilitou sua compreensão da aplicação e reduziu a necessidade de uma explicação prévia extensiva.

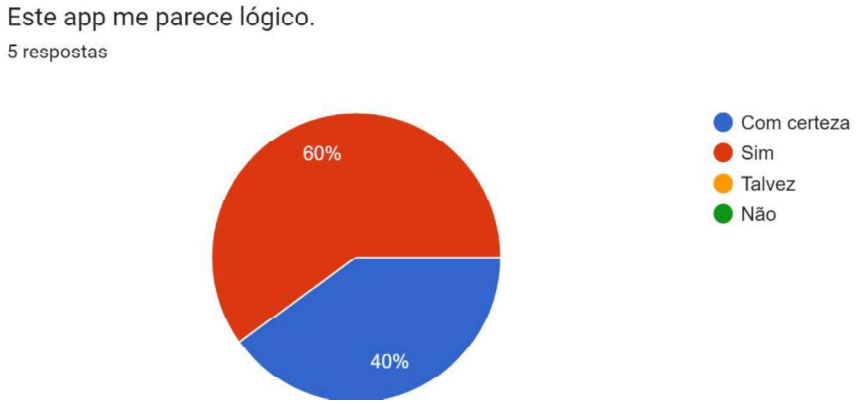
3.2 QUESTIONÁRIO DE AVALIAÇÃO

Após a finalização dos testes de usabilidade, os participantes responderam de forma anônima a um questionário de avaliação. Este questionário foi desenvolvido com base no WAMMI (*Website Analysis and Measurement Inventory*), visando permitir que os participantes compartilhassem suas experiências com a aplicação.

As questões foram disponibilizadas por meio do *Google Forms*, uma ferramenta do *Google* para a criação de formulários *online* voltados à coleta de dados. Ao término da pesquisa, a plataforma facilita a análise apresentando os resultados em gráficos, permitindo uma visualização clara e imediata das respostas.

A ilustração 23 apresenta as respostas dos usuários em relação à lógica do sistema. Do total, 60% afirmaram que "Com certeza" consideraram a aplicação lógica, enquanto os outros 40% responderam que "Sim", reforçando a ideia de que a estrutura e funcionamento da ferramenta foram compreendidos.

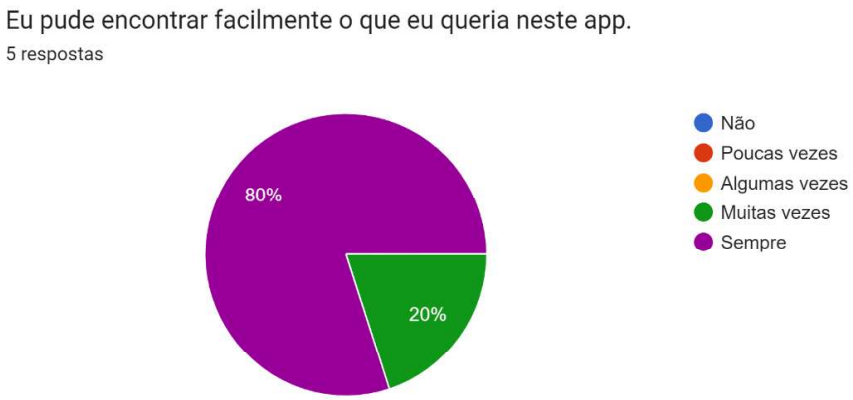
Ilustração 23 - Gráfico de respostas da primeira pergunta



Fonte: Autores, 2024

A ilustração 24 apresenta as respostas dos usuários para a afirmação “Eu pude encontrar facilmente o que eu queria neste *app*”. O gráfico indica que 80% dos participantes afirmaram “Sempre” e 20% “Muitas vezes” terem encontrado o que buscavam durante o teste, sugerindo uma boa navegabilidade do sistema.

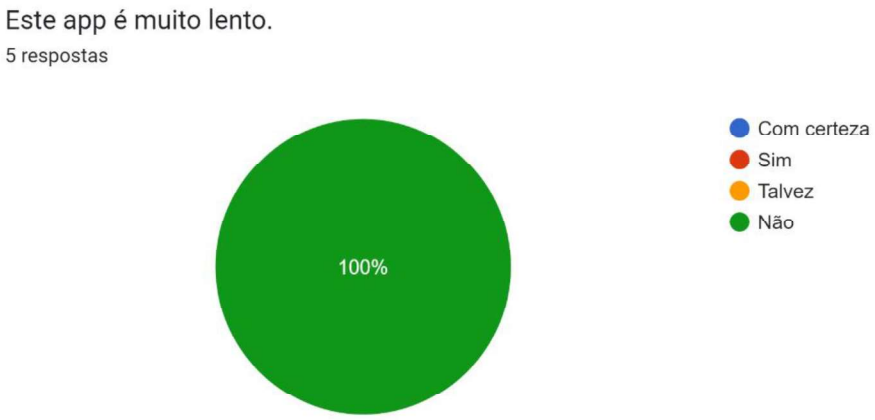
Ilustração 24 - Gráfico de respostas da segunda pergunta



Fonte: Autores, 2024

A ilustração 25 apresenta as respostas dos usuários em relação à lentidão do sistema, com uma avaliação totalmente negativa, conforme o esperado. Do total, 100% dos participantes afirmaram que “Não” consideraram a aplicação lenta durante seu uso, demonstrando uma baixa latência do sistema.

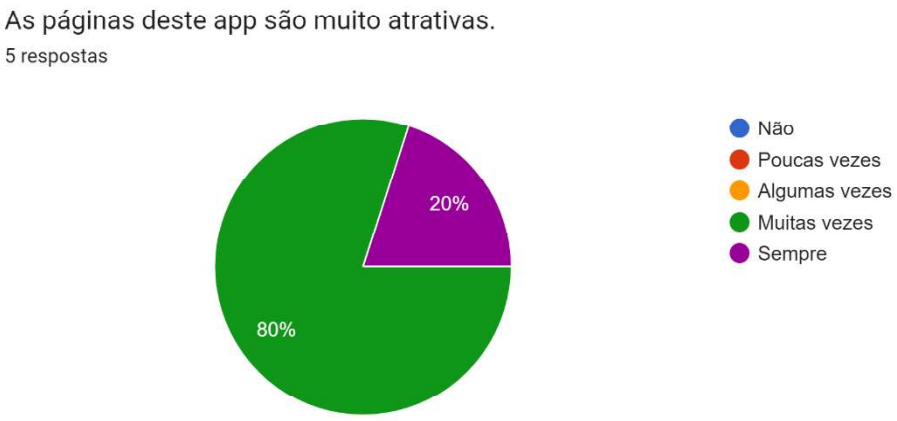
Ilustração 25 - Gráfico de respostas da terceira pergunta



Fonte: Autores, 2024.

As ilustrações 26 e 27 apresentam as respostas dos usuários em relação ao design do CalendAll.

Ilustração 26 - Gráfico de respostas da quarta pergunta

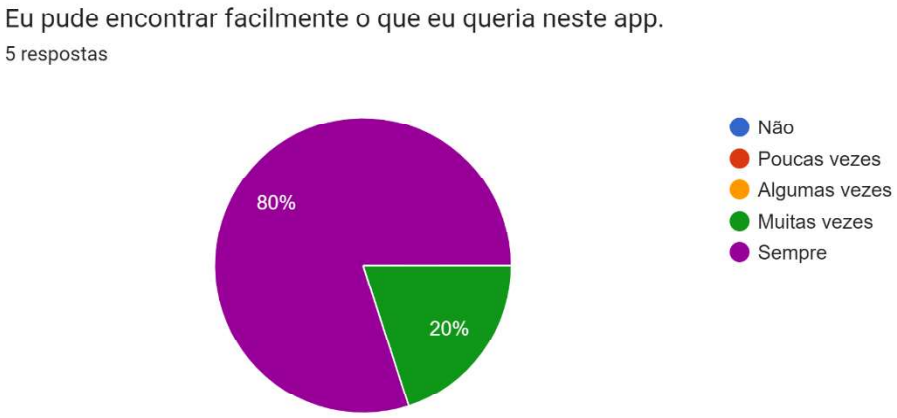


Fonte: Autores, 2024.

As respostas dos usuários para a afirmação “As páginas deste *app* são muito atrativas” estão presentes na ilustração 26 apresentada acima. O gráfico indica que 80% dos participantes afirmaram “Muitas vezes” e 20% afirmaram “Sempre” considerarem as páginas da aplicação atrativas.

Ilustração 27 - Gráfico de respostas da quinta pergunta

.



Fonte: Autores, 2024.

O gráfico apresentado na ilustração 27 demonstra as respostas dos usuários à afirmação “Eu pude encontrar facilmente o que eu queria neste *app*”. É indicado que 80% dos participantes afirmaram que “Sempre”, enquanto 20% afirmaram que “Muitas Vezes”, puderam encontrar facilmente o que queriam na aplicação.

Sendo assim, é possível compreender uma avaliação positiva quanto à apresentação do sistema, demonstrando que o *layout* e a estrutura das informações foram eficazes para proporcionar uma boa experiência do usuário, com telas atrativas, intuitivas e de fácil entendimento.

3.3 CONCLUSÃO

O CalendAll foi projetado com o principal objetivo de facilitar a organização acadêmica dos estudantes universitários, oferecendo personalização de notificações, definição de prioridades para eventos baseado em cores e estilos visuais diferentes, e um ambiente colaborativo onde alunos podem criar salas para compartilhar eventos

coletivos, ajudando a minimizar problemas recorrentes como a procrastinação e o esquecimento.

Após a realização dos testes, constatou-se que o desenvolvimento da aplicação atingiu as expectativas e os objetivos iniciais, apresentando uma avaliação positiva quanto ao design e funcionamento do sistema.

No entanto, há espaço para futuros aprimoramentos no CalendAll. Em versões futuras, será integrado aos sistemas do *Microsoft Teams* e *Google Classroom*, de forma que os eventos adicionados em turmas destas outras aplicações sejam automaticamente sincronizados com o calendário pessoal do aplicativo. Além disso, o uso de uma inteligência artificial generativa para auxiliar os estudantes em sua organização acadêmica e em criação de materiais de apoio de acordo com o conteúdo da tarefa, será um diferencial para tornar a experiência ainda mais intuitiva e eficiente.

REFERÊNCIAS

BRASIL. Instituto Justiça e Segurança Nacional. Boletim Educação: Tendências e Desafios. Boletim da educação PNAD contínua. **Instituto Jones dos Santos Neves**, [S.l], 2023. Disponível em: https://ijsn.es.gov.br/Media/IJSN/PublicacoesAnexos/boletins/Boletim_Educacao_2023T1.pdf. Acesso em: 03 abr. 2024.

DANIELLSON, William. *React Native application developmen: A comparison between native Android and React Native.* ***Linköping University***, 2016. Disponível em: https://liu.diva-portal.org/smash/record.jsf?pid=diva2%3A998793&dswid=5335. Acesso em 04 out. 2024.

GOOGLE. O que é e para que serve o Google Sala de Aula?. **Google for Education**, [S.l.], s.d. Disponível em: https://edu.gcfglobal.org/pt/google-sala-de-aula-para-alunos/o-que-e-e-para-que-serve-o-google-sala-de-aula/1/. Acesso em: 02 abr. 2024.

JUNIOR, Edemilton Alcides Galindo; ROCHA, Romeu Dias; MACIEL, Ronierison de Souza. Desenvolvimento de API REST com Spring Boot. **Revista Científica do UniRios**, Rio de Janeiro, 2021. Disponível em: http://www.publicacoes.unirios.edu.br/index.php/revistarios/article/view/102/102. Acesso em 16 set. 2024.

MICROSOFT. Introdução ao Microsoft Teams. **Suporte da Microsoft**, [S.l.], s.d. Disponível em: https://support.microsoft.com/pt-br/office/introdu%C3%A7%C3%A3o-ao-microsoft-teams-b98d533f-118e-4bae-bf44-3df2470c2b12. Acesso em: 13 abr. 2024.

NIELSEN, Jakob. *Usability Engineering*. ***Academic Press,*** San Diego, 1993. Acesso em 15 out. 2024.

ORACLE. Java. [S.l.] , s.d. Disponível em: https://www.oracle.com/br/java/. Acesso em 05 out. 2024.

PUC MINAS. Estudar e trabalhar: como organizar sua semana para ter um bom desempenho. **Conexão PUC Minas**, Belo Horizonte, 6 set 2023. Disponível em: https://conexao.pucminas.br/blog/dicas/estudar-e-trabalhar/. Acesso em: 03 abr. 2024.

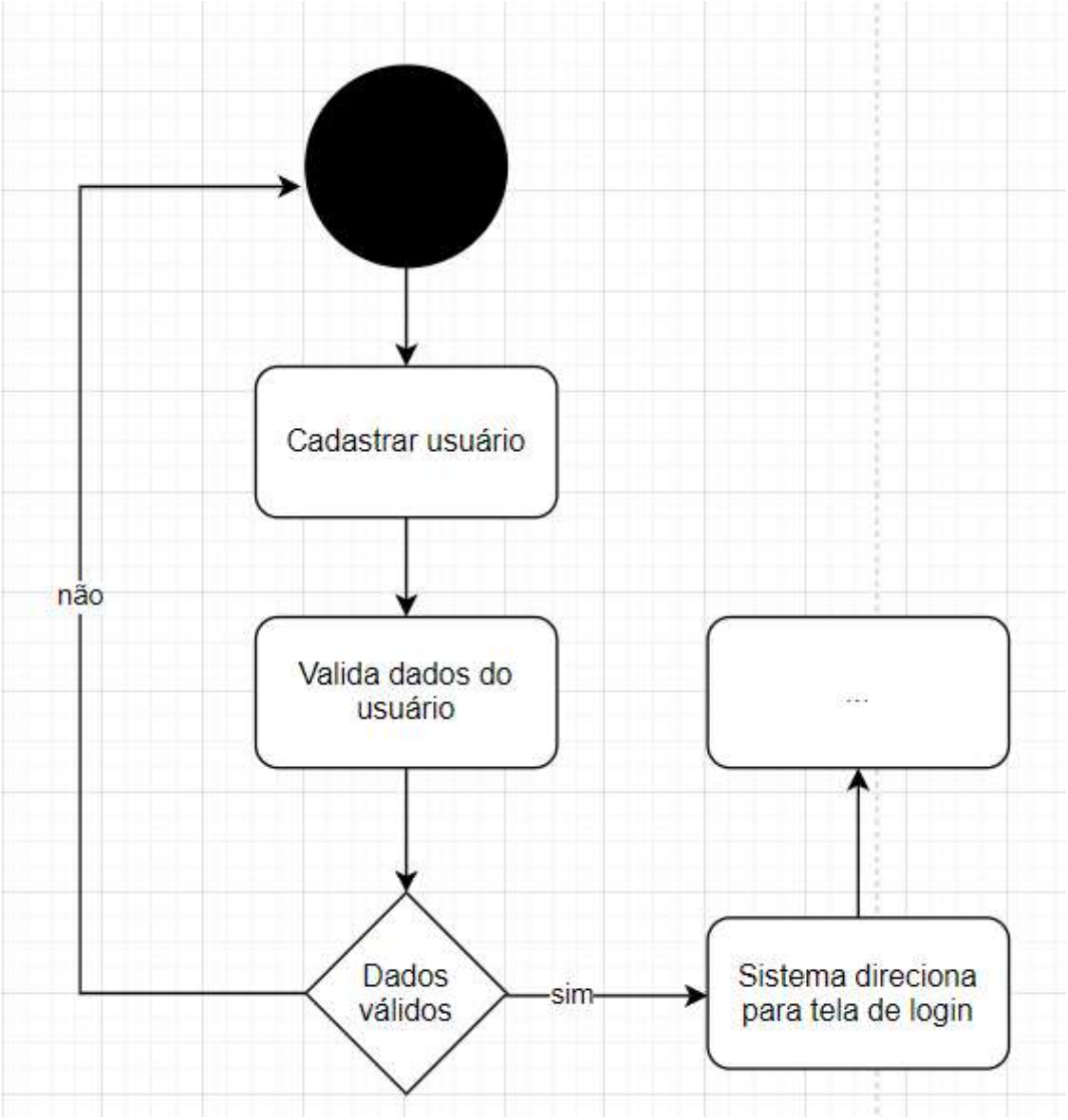
SANTOS, Vieira Joene; MALAQUIAS, Vivian Nathalia Rodrigues. Procrastinação acadêmica entre estudantes universitários brasileiros. **Educação em Foco**, Belo Horizonte, v. 25, n. 1, p. 1-22, jan.-abr. 2023. Disponível em: https://revista.uemg.br/index.php/educacaoemfoco/article/view/5816/4414. Acesso em: 14 mar. 2024.

SPRING. *Spring Boot Overview*. [S, l], s, d. Disponível em: https://docs.spring.io/spring-boot/index.html. Acesso em: 04 out. 2024.

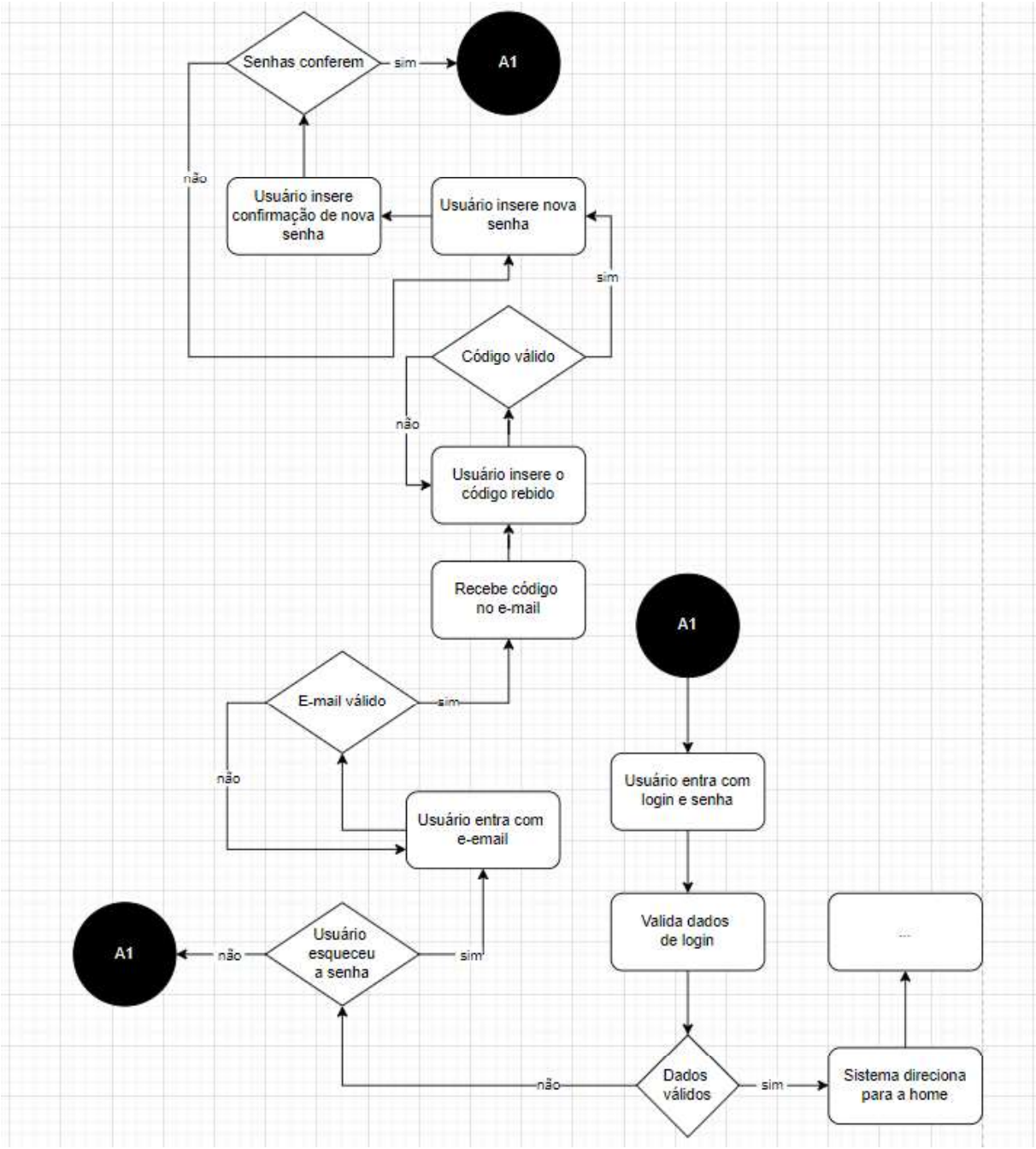
YOSHIY, Shimeny Michelato; KIENEN, Nádia. Gerenciamento de Tempo: Uma Interpretação Analítico-Comportamental. **Psic. da Ed.**, São Paulo, 47, 2º sem. de 2018, pp. 67-77. Disponível em: n47a08.pdf (bvsalud.org). Acesso em: 20 mar. 2024.

APÊNDICE A

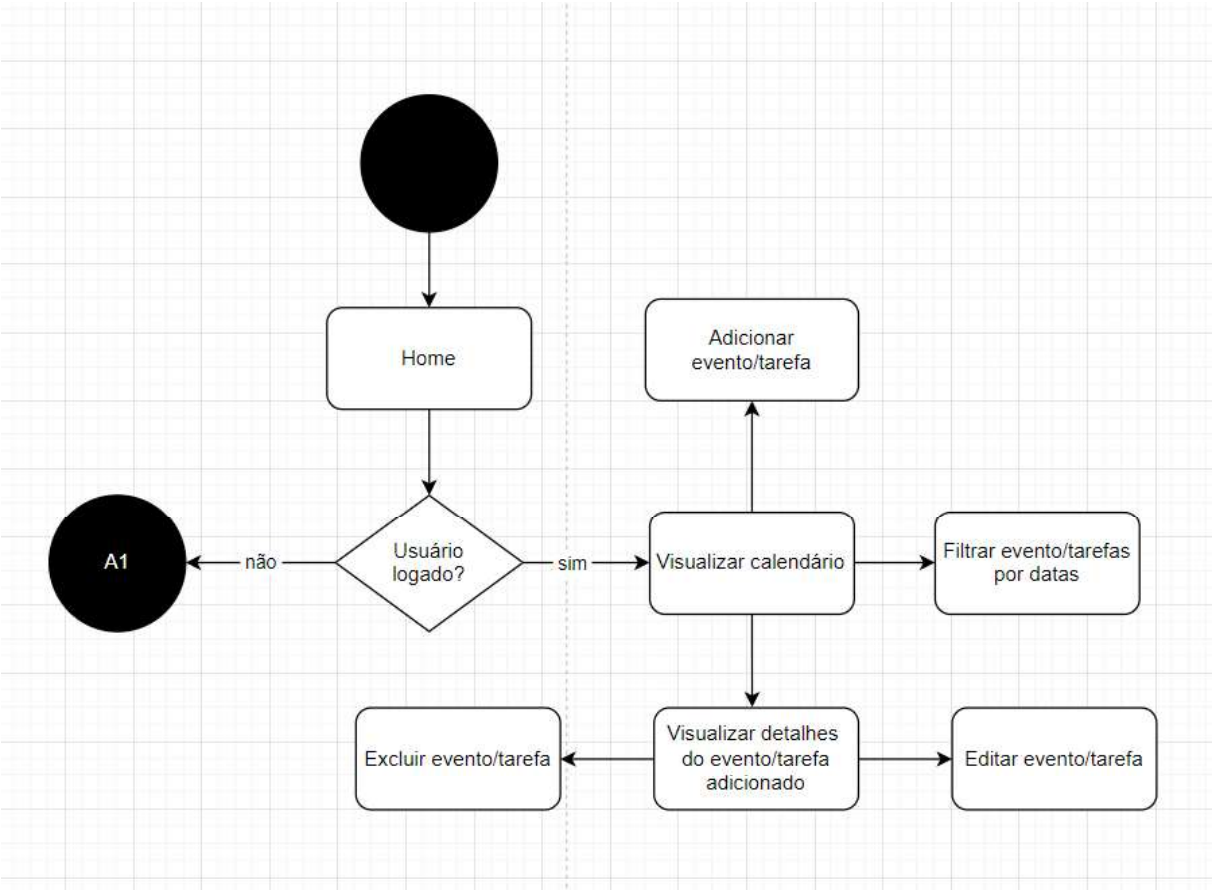
Cadastro de usuário



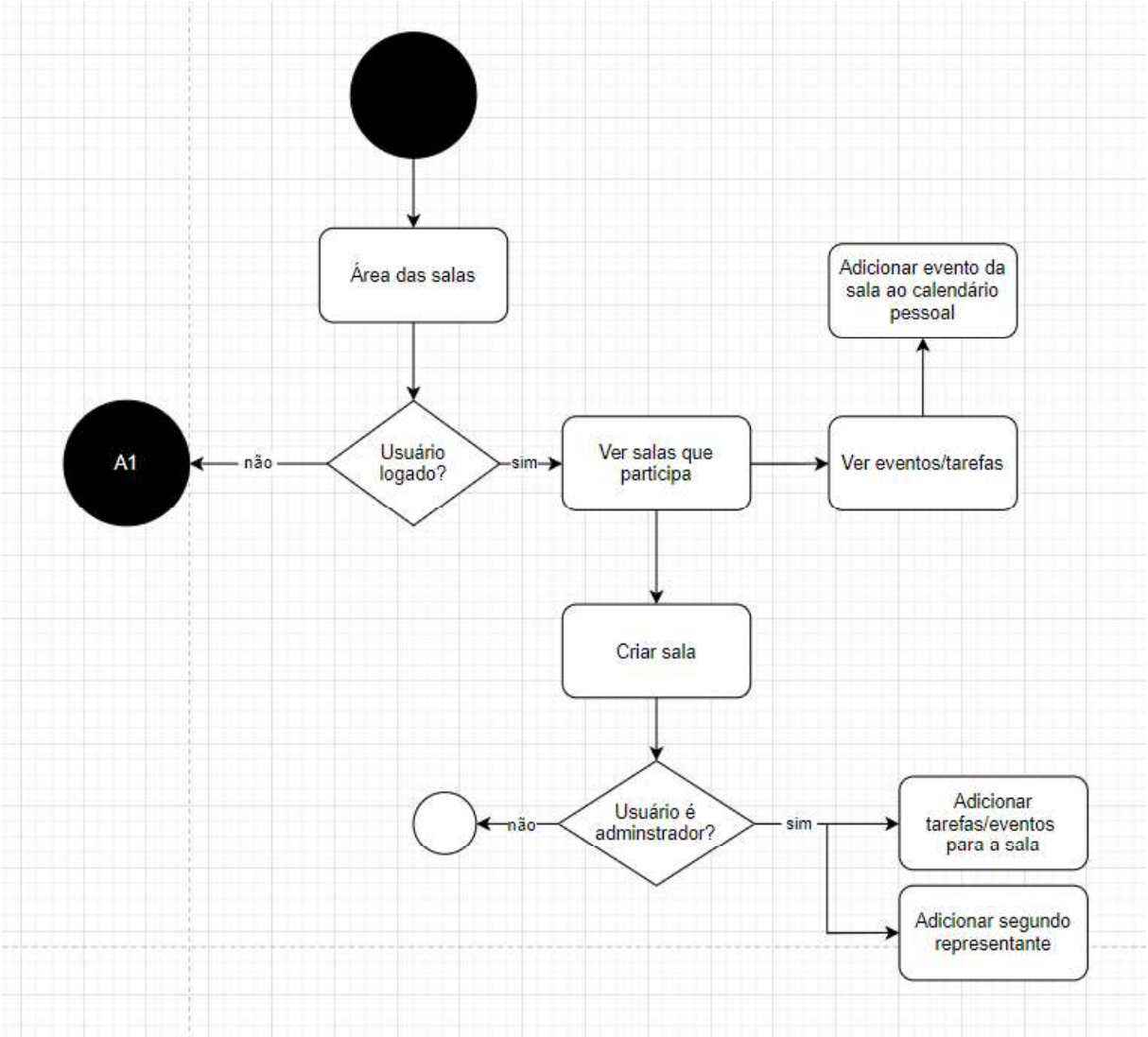
Login e recuperação de senha



Home


Home
Adicionar evento/tarefa
A1
não
Usuário logado?
sim
Visualizar calendário
Filtrar evento/tarefas por datas
Excluir evento/tarefa
Visualizar detalhes do evento/tarefa adicionado
Editar evento/tarefa

Área das salas



Configurações