



Sistema de Comunicação Distribuído

Daniel Terra Gomes

Ausberto S. Castro Vera

UENF - CCT - LCMAT - CC

5 de novembro de 2022

Copyright © 2022 Ausberto S. Castro Vera e Daniel Terra Gomes

UENF - UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO

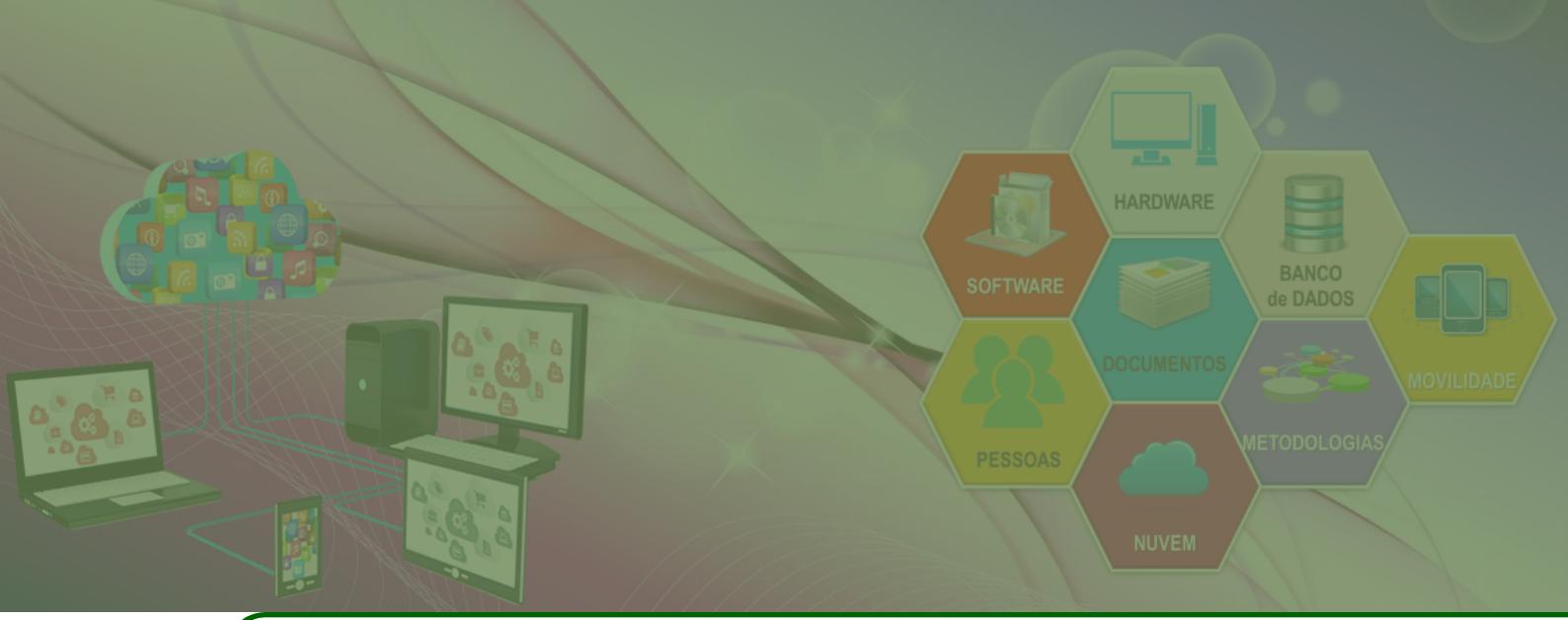
CCT - CENTRO DE CIÊNCIA E TECNOLOGIA
LCMAT - LABORATÓRIO DE MATEMÁTICAS
CC - CURSO DE CIÊNCIA DA COMPUTAÇÃO



Sumário

1	Introdução	1
1.1	Escopo ou Contextualização do Sistema OO	2
1.1.1	Sistema	2
1.1.2	Sistema de gerenciamento de conversas	2
1.2	Objetivo do Sistema	2
1.3	Justificativa	3
2	Requisitos do Sistema OO	5
2.1	Requisitos Funcionais	5
2.1.1	Subsistema do usuário	5
2.1.2	Subsistema de Agendas	6
2.1.3	Subsistema de Login	6
2.1.4	Subsistema de elaboração de relatórios	6
2.1.5	Subsistema de Acesso ao Sistema	6
2.2	Requisitos Não-Funcionais	6
2.2.1	Requisitos de Usabilidade	6
2.2.2	Requisitos de Confiabilidade	7
2.2.3	Requisitos de Disponibilidade	7
2.2.4	Requisitos de Privacidade	7
2.2.5	Requisitos de Acesso	7
2.3	Requisitos de Negócios	8

3	Modelagem do Sistema	9
3.1	Diagramas DFD	9
3.1.1	Diagramas DFD Login - entidade externa	9
3.1.2	Diagramas DFD Login - entidade interno - Login	10
3.2	Diagramas E-R	10
3.3	Diagramas de Classes	11
3.4	Diagramas Casos de uso	11
3.4.1	Cadastro dos usuários	11
3.4.2	Criação de grupo	11
3.4.3	Excluir grupo	11
3.4.4	Diagrama de Caso de Uso	11
3.5	Diagramas Sequência	13
3.6	Diagramas de Atividades	14
3.7	Diagramas Estado	16
4	Projeto do Sistema OO	17
4.1	Arquitetura do Sistema - Classes	17
4.2	Interfaces do Usuário	17
4.3	Tabelas de Dados	22
5	Implementação do Sistema OO	23
5.1	Programação	23
5.1.1	Classes	23
5.1.2	Código Fonte	25
5.1.3	Banco da Dados	26
5.1.4	Código Fonte - About Us	27
5.2	Documentação do Software	28
6	Considerações Finais	29
	Bibliografia	31



1. Introdução

A *Kalender* é um sistema que tem como objetivo possibilitar o agendamento de tarefas de seu usuário. O processo de cadastro de atividades possibilita que o usuário mantenha um histórico das suas atividades diárias. Um calendário é um gráfico ou dispositivo que exibe a data e o dia da semana e, muitas vezes, todo um ano específico dividido em meses, semanas e dias. Dessa maneira, possibilitamos os usuários do site a manter suas atividades sem perder os horários ali cadastrados. Nossa missão é ser um calendário online que permite que um ou mais usuários editem e, opcionalmente, compartilhem com outros usuários o acesso online a um calendário.

A partir dos nossos serviços poderá fazer uso de uma ferramenta indispensável. Pois, nossas vidas diárias são gerenciáveis apenas graças aos nossos calendários em nossos computadores e telefones. Um calendário online permite-nos gerir os nossos horários e ajuda-nos na gestão do tempo. Pense em todas as vezes que as pessoas se esquecem de suas reuniões e compromissos agendados. As consequências incluem remarcar as reuniões para uma data muito posterior ou até mesmo pagar uma taxa por faltar a um compromisso. Um calendário online permite que você sempre acompanhe suas reuniões e compromissos.

A empresa propõe que seus serviços sejam disponibilizados para todas as cidades do país 24 horas por dia, conectando pessoas que necessitam se comunicar pela internet. Portanto, sendo Sistema Distribuído uma coleção de sistemas de computador autônomos que são fisicamente separados, mas conectados por uma rede de computadores centralizada que é equipada com software de sistema distribuído. Os computadores autônomos se comunicarão entre cada sistema compartilhando recursos e arquivos e executando as tarefas atribuídas a eles. Após a conclusão bem-sucedida dessas solicitações, o sistema irá conectar os usuários que solicitaram a conexão [dev22]. As referências bibliográficas básicas deste projeto são: [DWR14], [Hel13], [Gue11], [Som18] e [Waz11]. Como bibliografia complementar serão considerados: [SJB12], [SR12] e [Fur13]. Esses mesmos serviços da empresa só são desligados quando há necessidade de manutenção, medidas de precaução e suprimentos. Este sistema é gerenciado remotamente por desenvolvedores e colaboradores sem a necessidade de colaboradores em um local central. Esses funcionários se comunicam por meio de um sistema interno projetado para manter a comunicação segura e rápida entre os funcionários.

Além disso, o desenvolvimento deste projeto também visa conectar os serviços da empresa. Esses serviços serão desenvolvidos por uma empresa terceirizada que já os projetam para receber o software de inteligência artificial desenvolvido desde o início pela empresa desenvolvedora do sistema de suporte inteligente. A conexão desses serviços funcionará a partir de torres de comunicação que estão espalhadas por todo o Brasil e essas torres são conectadas a satélites espalhados por todo o planeta.

Dessa forma, o acesso a satélites de comunicação e recursos em nuvem é contratado por empresas terceirizadas. É importante ressaltar que os recursos de projeto, desenvolvimento e manutenção desses serviços são provenientes de empresas terceirizadas.

Este projeto tem como foco a implantação desta tecnologia no Brasil. Aqui, entre outras coisas, são apresentados seus requisitos, recursos, casos de uso, componentes, orçamentos para viabilizar a realização deste sistema.

Neste primeiro capítulo de abertura descreve, e elabora o projeto a ser desenvolvido.

1.1 Escopo ou Contextualização do Sistema OO

Esta seção apresenta informações sobre o sistema, prioridades e justificativas para atingir seus objetivos gerais de facilitar a organização de tarefas entre os serviços da empresa, seus usuários, e facilitar as transações. A partir de um sistema que agiliza o processo de comunicação dos serviços da empresa e encaminha esses recursos para o usuário final.

1.1.1 Sistema

Um dos principais objetivos do projeto é oferecer aos usuários uma agilidade no processo de organização das suas atividades, esse serviço será desenvolvido seguindo princípios da *Programação Orientada a Objetos*, onde possam ter uma boa experiência com um sistema mais inteligente. Dessa forma é possível, por meio do sistema, solicitar o conexões de usuários mais próximo que atenda às necessidades do usuário, bem como planejar atividades, consultar atividades anteriores, dar feedback sobre o serviço prestado pela empresa e se cadastrar no sistema.

1.1.2 Sistema de gerenciamento de conversas

Sistema de agenda dos serviços e gestão dos recursos da empresa; As agendas. Desta forma, propõe-se o desenvolvimento de um sistema que permita esta integração, de forma a facilitar e agilizar os processos diários. Este sistema também visa proporcionar à empresa um ambiente de conexão remoto pensado para manter a agilidade segura e rápida entre os usuários. Além disso, podem ser consultadas informações sobre as atividades dos usuários, como tempo de uso, manutenção, tarefas já realizadas, localizações, status, tarefas atuais, contatos, pendências.

1.2 Objetivo do Sistema

O objetivo geral do projeto, é oferecer recursos para o usuário, na qual possa proporcionar uma boa experiência com o sistema. Será possível através do sistema, realizar o cadastro e visualizar atividades.

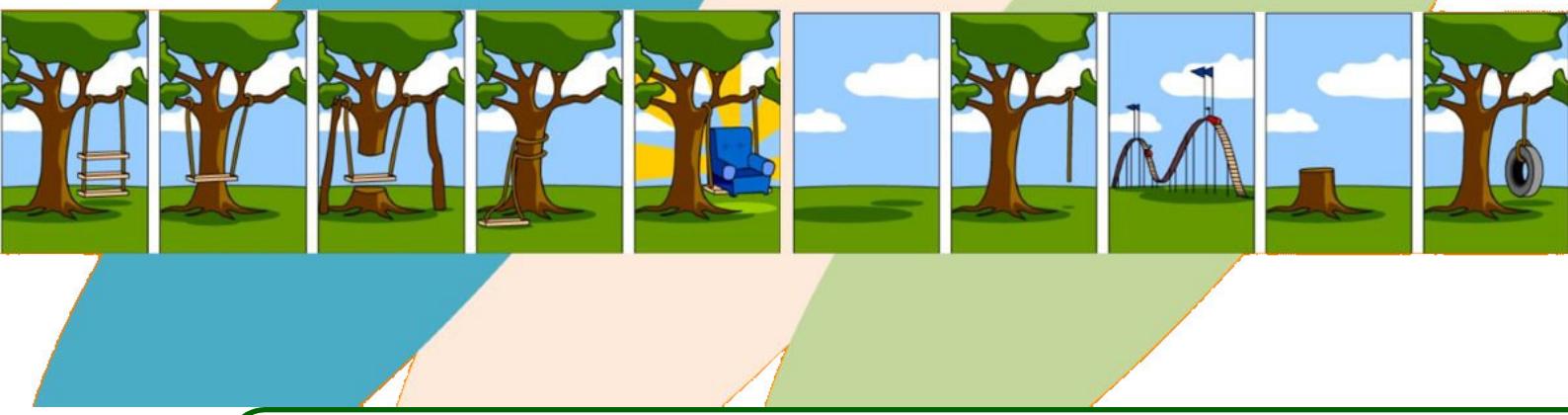
- **Objetivos específicos:**

- Adicionar atividades em agenda e será possível ser visualizado posteriormente.

- Permitir realizar cadastro de novas atividades no sistema, e visualizar as de outros usuários presentes no sistema.
- Permitir realizar o login e cadastro de usuários.

1.3 Justificativa

O sistema visa buscar uma descentralização das atividades de maneira virtual para que os calendários possam ser acessados por todos os 27 estados brasileiros. Dessa forma, é possível, por meio do sistema, solicitar as conexões de usuários mais próximas que atendam às necessidades do usuário, bem como planejar atividades, consultar atividades anteriores, dar feedback sobre as tarefas prestadas pela empresa e se cadastrar no sistema. Este sistema é gerenciado remotamente por desenvolvedores e colaboradores sem a necessidade de colaboradores em um local central. Esses funcionários se conectam por meio de um sistema interno projetado para manter a comunicação segura e rápida entre os funcionários, a partir de um sistema que agiliza o processo de atividades dos serviços da empresa e encaminha esses recursos para o usuário final. Desse modo, devido às necessidades de gerenciamento de atividades presentes no país. Será desenvolvido o sistema proposto anteriormente.



2. Requisitos do Sistema OO

O Requisito do Sistema inclui um conjunto de tarefas que devem ser executadas para criar uma documentação de requisitos como um produto final. Tudo o que consta nos documentos permite que o software seja criado, atualizado e reparado sempre que necessário, de acordo com o estabelecido inicialmente.

Neste capítulo é apresentado listas, definições e especificações de Requisitos do sistema ser desenvolvido. Os requisitos são declarações abstratas de alto nível sobre os *serviços* que o sistema deve prestar à organização, e as *restrições* sobre as quais deve operar. Os requisitos sempre refletem as necessidades dos clientes do sistema.

Sobre os requisitos, Raul S. Wazlawick afirma:

A *etapa de levantamento de requisitos* corresponde a buscar todas as informações possíveis sobre as funções que o sistema deve executar e as restrições sobre as quais o sistema deve operar. O produto dessa etapa será o documento de requisitos, principal componente do anteprojeto de software.

A *etapa de análise de requisitos* serve para estruturar e detalhar os requisitos de forma que eles possam ser abordados na fase de elaboração para o desenvolvimento de outros elementos como casos de uso, classes e interfaces.

O levantamento de requisitos é o processo de descobrir quais são as *funções* que o sistema deve realizar e quais são as *restrições* que existem sobre estas funções [Waz11].

2.1 Requisitos Funcionais

Como parte da fase de coleta, requisitos funcionais são todos os problemas e necessidades que precisam ser atendidos e resolvidos pelo software por meio de recursos ou serviços. Nesta seção trataremos dos requisitos funcionais:

2.1.1 Subsistema do usuário

1. Cadastro de novos usuário;
2. Edição de informações cadastrais;

3. Deletar do usuário;
4. Gerenciamento de mensagens do usuário;
5. Gerenciamento de grupos criados pelo usuário.

2.1.2 Subsistema de Agendas

1. Criar agenda;
2. Edição de informações da agenda;
3. Exclusão de atividade;
4. Listagem de pessoas da atividade;
5. Mensagem para todos os membros da atividade.

2.1.3 Subsistema de Login

1. Realizar a verificação dos dados do usuário ao login;
2. Gerar token de autenticação;
3. Permitir acesso funcionalidade apenas a usuários habilitados;
4. Realizar cadastro de novas credenciais de usuários;
5. Verificar se o usuário já está cadastrado;
6. Permitir troca de senha de usuários cadastrados.

2.1.4 Subsistema de elaboração de relatórios

1. O sistema deve permitir o registro e atualização de relatórios no sistema;
2. Cada relatório só pode ser atribuído a uma usuário;
3. Cada relatório só pode ser atribuído a um grupo;
4. Relatório pode ser visto por todos os membros da agenda.

2.1.5 Subsistema de Acesso ao Sistema

1. Somente usuários cadastrados podem acessar o sistema;
2. O sistema deve permitir o acesso via e-mail;
3. O sistema deve fornecer um recurso de recuperação de senha;
4. O registro de um usuário não pode ser realizado se já houver um usuário cadastrado com os mesmos dados de e-mail.

2.2 Requisitos Não-Funcionais

Requisitos não funcionais são todos os requisitos relacionados a como o software implementa a realidade pretendida. Enquanto os requisitos funcionais se concentram no que é feito, os requisitos não funcionais descrevem como é feito.

Assim, todos os requisitos de sistema, hardware, software e operacionais são documentados separadamente [DWR14].

2.2.1 Requisitos de Usabilidade

1. O sistema deve fornecer uma interface fácil de navegar;

2. O sistema deve atingir um nível de desempenho satisfatório em sistemas leves;
3. O software deve realizar as ações com o mínimo de interações;
4. A interface deve se adaptar a diferentes dispositivos;
5. A implementação da interface deve seguir um padrão para garantir consistência;
6. A interface deve ser esteticamente agradável ao usuário.

2.2.2 Requisitos de Confiabilidade

1. Mantenha sempre os dados atualizados;
2. Tolerância a falhas do sistema;
3. Prevenção de falhas;
4. Manipulação de erros;
5. Em caso de erro ou falha, o sistema deve ser capaz de recuperar os dados;
6. O sistema deve notificar o usuário quando uma operação foi bem-sucedida ou falhou;
7. Todos os subsistemas de software devem ser monitorados.

2.2.3 Requisitos de Disponibilidade

1. O sistema deve estar sempre disponível para o usuário, exceto em dias de manutenção;
2. Durante a manutenção, o sistema pode ficar offline indefinidamente;
3. O sistema deve ser capaz de receber mais de 50 solicitações por minuto sem criar instabilidade;
4. A manutenção pode ser realizada conforme necessário.

2.2.4 Requisitos de Privacidade

1. Os dados insensíveis do usuário não precisam ser criptografados;
2. Garantir que os dados do usuário sejam usados apenas na plataforma;
3. Os dados do usuário só são coletados pela plataforma com a devida permissão do utilizador;
4. Os contatos do usuário são coletados apenas com o consentimento do usuário;
5. O sistema deve respeitar as leis de proteção de dados da região em que é utilizado;
6. O sistema pode permitir o cache de senhas;
7. As informações transmitidas podem não estar protegidas por técnicas de criptografia;
8. Nenhum dado pode ser compartilhado sem a permissão do proprietário;
9. O sistema pode coletar dados que não são críticos para o negócio.

2.2.5 Requisitos de Acesso

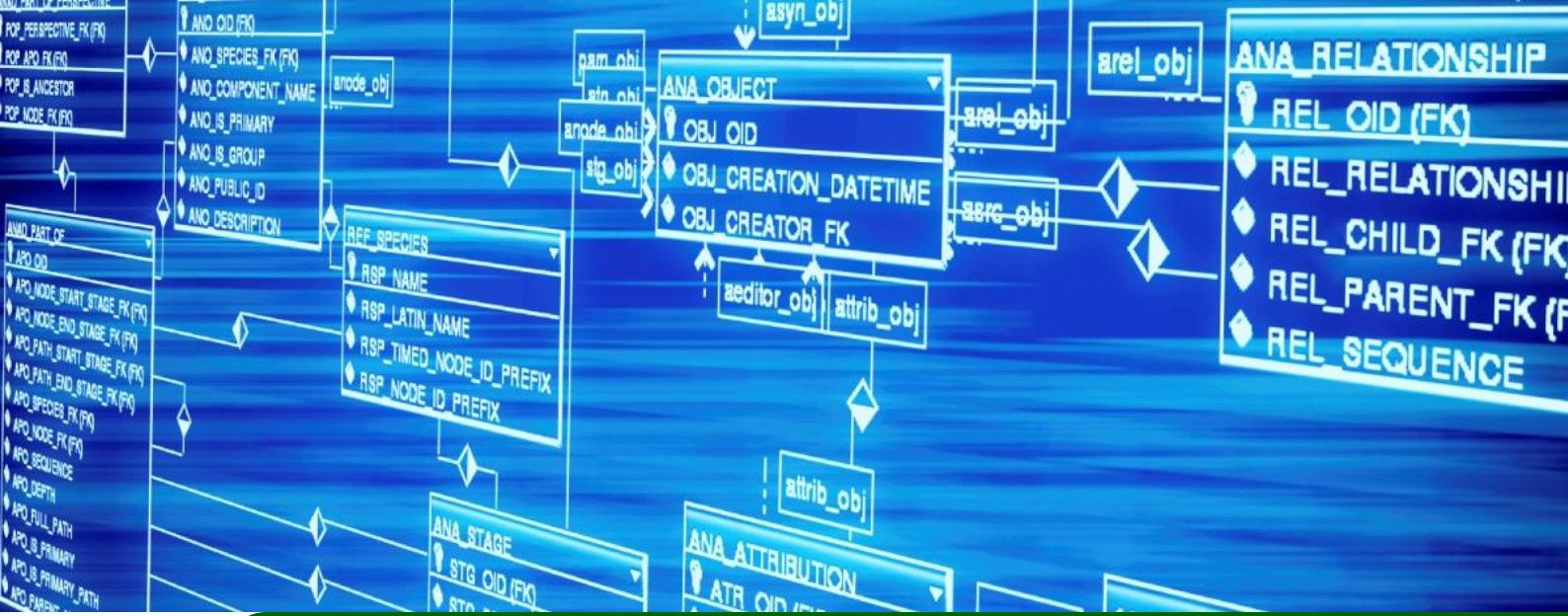
1. Somente usuários cadastrados e autenticados ou com link de acesso devem acessar a plataforma;
2. Garanta que as agendaprivados sejam acessados apenas por usuários autenticados;
3. Certifique-se de que os grupos sejam editados apenas pelo usuário da agenda;
4. Os grupos só podem ser excluídos pelo usuário moderador;
5. Todos os usuários autenticados podem marcar agendas.

2.3 Requisitos de Negócios

Requisitos do negócio são requisitos de alto nível que explicam e justificam qualquer projeto. Os requisitos de negócios são as atividades críticas de uma empresa que devem ser executadas para atender ao(s) objetivo(s) organizacional(is) enquanto permanecem independentes do sistema solução.

Para esse sistema os requisitos de negócios são os a seguir:

1. Facilitar a comunicação entre as pessoas em todo o país;
2. O sistema deve permitir que o usuário se conectem com outros usuários do sistema;
3. Fornecimento de funções de gestão de agendas;
4. Oferecer aos usuários o número de atividades na agenda;
5. Aumentar o número de usuários ativos na plataforma;
6. Melhore a segurança e a transparência dos dados e processos de negócios.



3. Modelagem do Sistema

Neste capítulo apresentamos diferentes perspectivas do sistema implementado usando diagramas. Os diagramas criados, também chamados de modelos, resumem os requisitos do sistema e os descrevem para o desenvolvedor.

A abstração, característica essencial dos modelos, permite compreender o contexto do sistema, suas funcionalidades e seus fluxos de dados sem se aprofundar nos detalhes de sua implementação.

- Diagrama de fluxo de dados: representação gráfica do fluxo de dados geral do sistema.
- Diagrama de Relacionamento de Entidade: Mostra o relacionamento entre entidades do sistema, como: B. a relação entre o usuário e a propriedade.
- Diagrama de Classes: Representa a estrutura de classes do sistema, mostrando os atributos e métodos de cada classe.
- Diagrama de Casos de Uso: Representação gráfica de casos de uso ou situações em que um objeto pode estar no sistema em um determinado momento.
- Fluxograma: Mostra a sequência de processos do sistema
- Diagrama de Atividades: Mostra o fluxo de controle das atividades do sistema.
- Diagrama de Estado: Representa a situação em que o sistema se encontra em um determinado momento ou durante o processo.

3.1 Diagramas DFD

O diagrama de fluxo de dados é um diagrama que mostra o fluxo de dados de um sistema e modela seus aspectos e processos.

3.1.1 Diagramas DFD Login - entidade externa

O diagrama de fluxo de dados do nível 1 do sistema possui 3 entidades: Registros de usuários, Login, Pesquisa de usuários.

- Cadastro de Usuário: Nesta entidade, o usuário preenche um formulário com dados pessoais

e senha, em seguida os dados são processados pelo sistema e armazenados no banco de dados.

- Login: O usuário insere as credenciais de acesso ao sistema e o sistema verifica se as credenciais são válidas. Se forem válidos, o sistema salva o token de acesso do usuário no armazenamento local, caso contrário, o sistema retorna uma mensagem de erro.
- Buscar Usuários: As informações do usuário são fornecidas nesta entidade, então o sistema procura os usuários que possuem as informações fornecidas

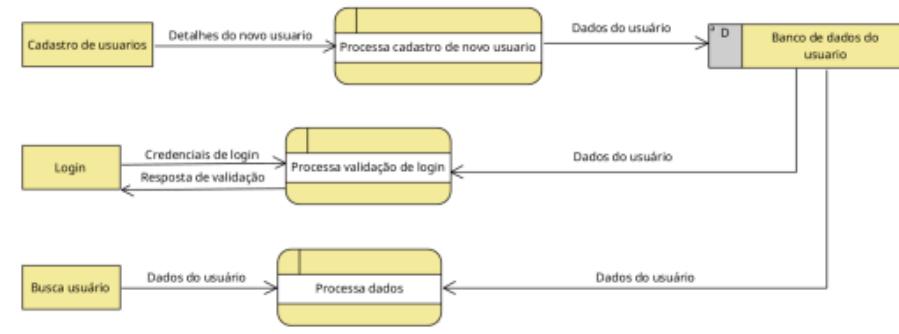


Figura 3.1: Fluxo de dados do sistema Cadastro

3.1.2 Diagramas DFD Login - entidade interno - Login

O diagrama de fluxo de dados de nível 2 mostrado é o subsistema de login que possui a entidade de usuário onde o endereço de email e a senha do usuário são fornecidos, então as informações são processadas pelo sistema e verificam se as credenciais são válidas.

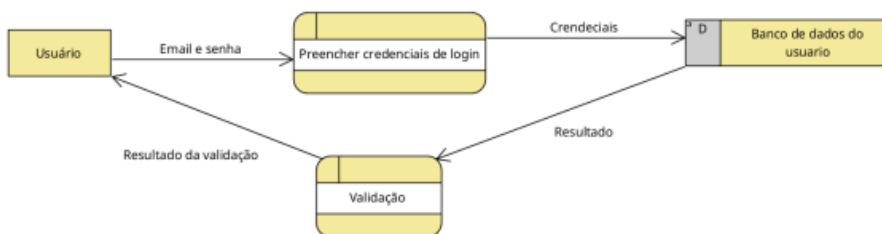


Figura 3.2: Fluxo de dados do sistema Login

3.2 Diagramas E-R

O diagrama E-R é a representação gráfica de uma entidade relacional, onde a entidade é representada por um objeto e os relacionamentos são representados por relacionamentos. O diagrama E-R do sistema completo é mostrado no exemplo dessa subseção.

O diagrama E-R do sistema de agenda possui as seguintes entidades: Usuário, agenda. Possui também os links: Usuários logado, onde o usuário pode cadastrar um ou mais agendas.

3.3 Diagramas de Classes

O diagrama de classes visa mostrar como o software está estruturado e como seus componentes estão interligados, ou seja, as operações que são realizadas entre eles.

3.4 Diagramas Casos de uso

Esta seção apresenta os casos de uso do sistema, cujo objetivo é mostrar as interações realizadas pelo usuário e as reações do sistema.

3.4.1 Cadastro dos usuários

1. Preenchimento dos dados cadastrais do usuários;
2. Login no sistema;
3. Verificação de dados;
4. Confirmação de dados;
5. Gerar informações do usuário;
6. Finalizar cadastro.

3.4.2 Criação de grupo

1. Cadastro de grupo no sistema;
2. Preenchimento de dados;
3. Informar valor do recurso para armazenamento;
4. Atualizar dados do sistema;
5. Finalizar cadastro de novos dados;
6. Gerar relatório atual.

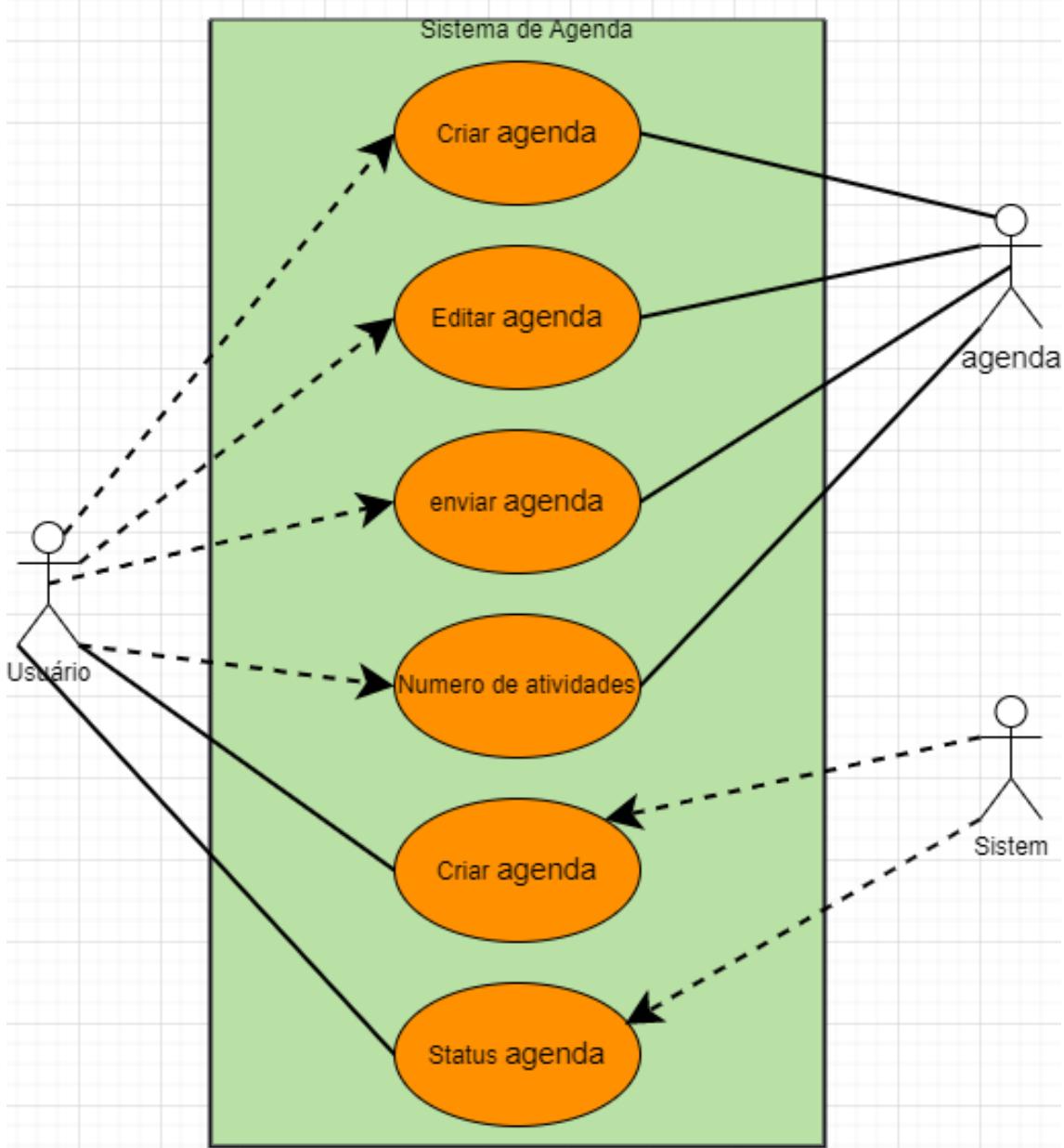
3.4.3 Excluir grupo

1. Atualizar dados do sistema;
2. Deletar grupo para todos os usuários.

3.4.4 Diagrama de Caso de Uso

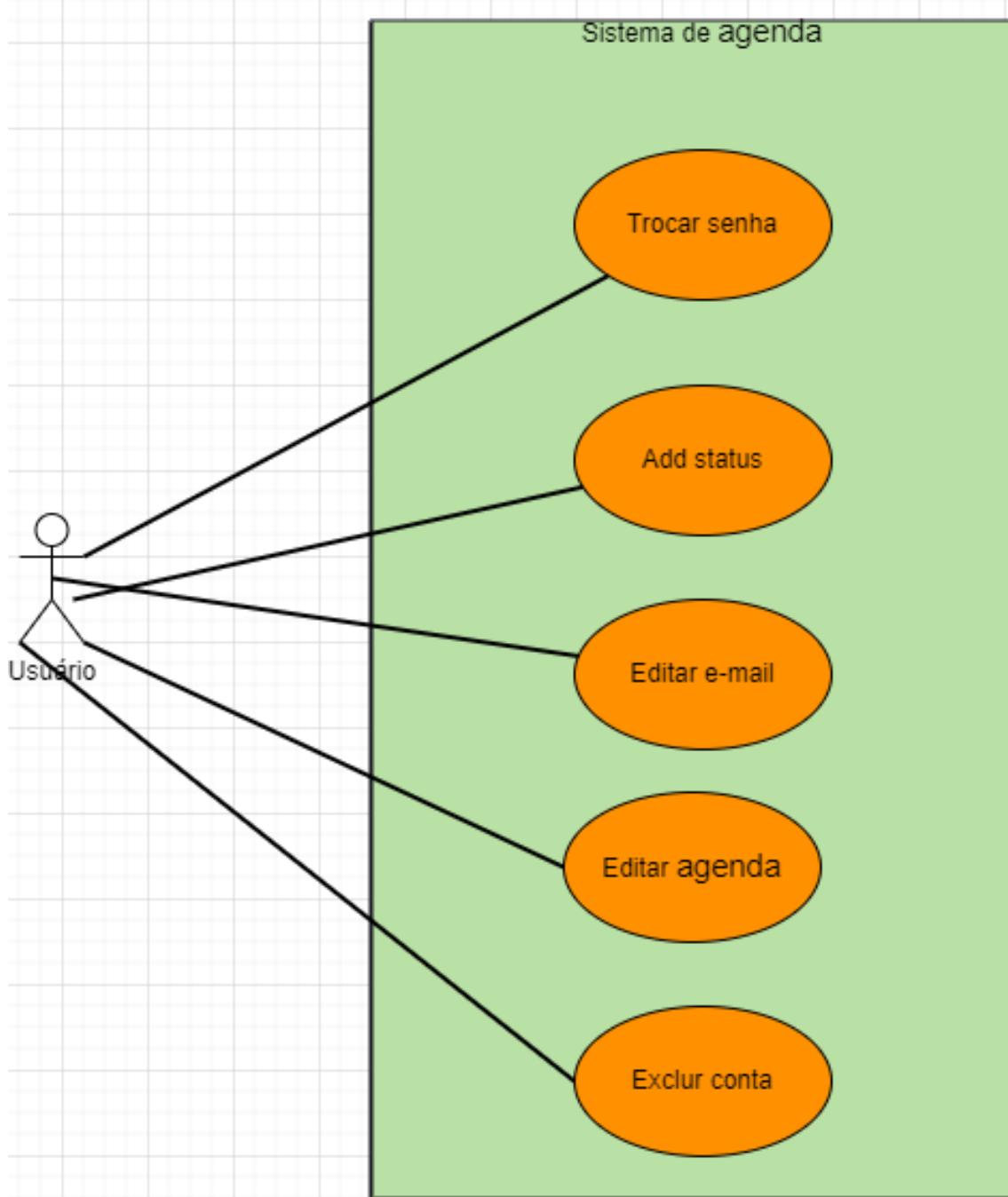
O diagrama a seguir pretende representar graficamente o caso de um "Serviço a um Usuário". Cujo tem como objetivo representar o caso de uso do sistema e exibir informações sobre ele.

Figura 3.3: Diagrama de caso de uso ao criar um grupo



No subsistema de usuário, você pode alterar os detalhes do usuário, alterar senhas, adicionar fotos de perfil e editar informações pessoais, como nome, e-mail e telefone. E também é possível excluir a conta do usuário.

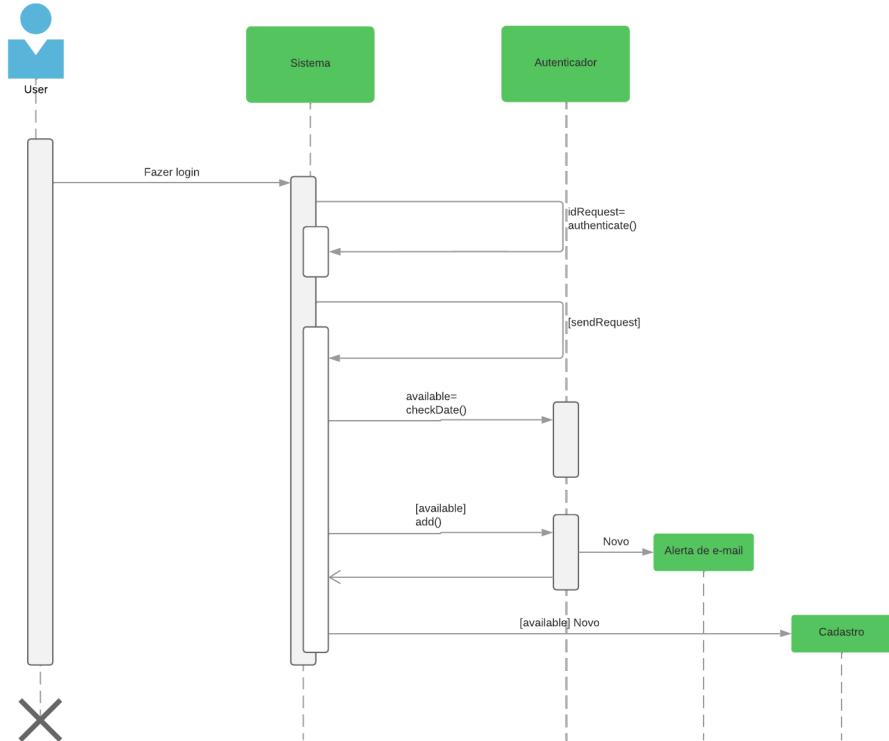
Figura 3.4: Diagrama de caso de uso de subsistema de usuário



3.5 Diagramas Sequência

O diagrama de sequência é uma solução de modelagem dinâmica popular em UML porque se concentra especificamente nas linhas de vida, ou nos processos e objetos que vivem ao mesmo tempo e nas mensagens trocadas entre eles para executar uma função antes que a linha de vida termine.

Figura 3.5: Diagrama de sequência de login

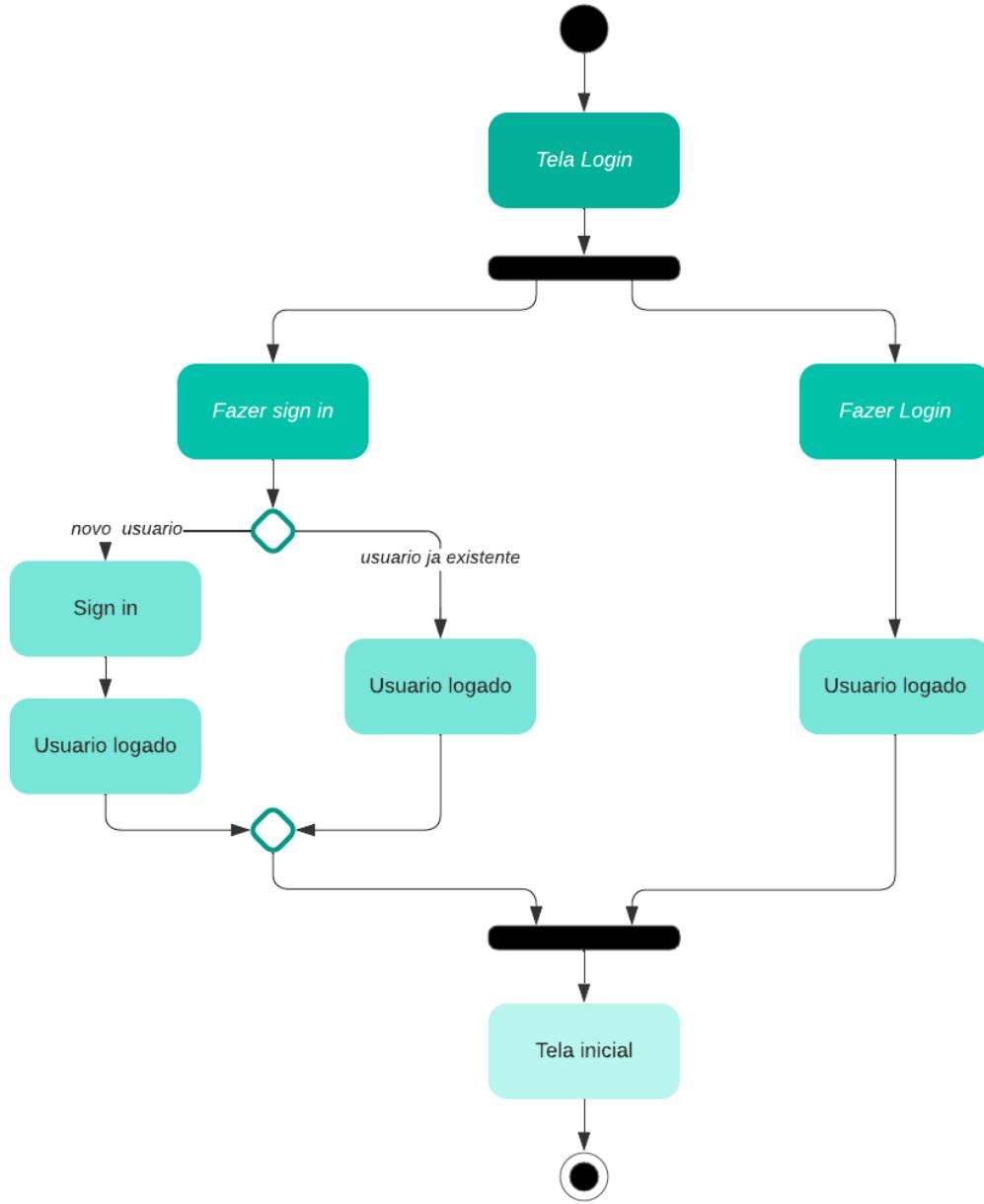


Primeiro, o usuário faz login com o endereço de e-mail e a senha, depois o autenticador se comunica com o sistema e verifica se o usuário existe. Nesse caso, o sistema cria uma sessão para o usuário e retorna o acesso de verificação ou token (um código gerado pelo sistema e usado para realizar operações no sistema). Por fim, mostrará que o usuário está logado, então ele será redirecionado para a página principal.

3.6 Diagramas de Atividades

Um diagrama de atividades é essencialmente um fluxograma que mostra as atividades realizadas por um sistema. Assim, ilustra o fluxo ou sequência de ações executadas em um sistema.

Figura 3.6: Diagrama de atividade de login



Primeiramente o usuário será levado para a tela de login onde será submetido a uma verificação se o usuário já está cadastrado, caso contrário o usuário será levado para a tela de cadastro do usuário, caso contrário será levado para a tela de login. Se o usuário estiver cadastrado, o usuário compartilhará o endereço de e-mail e a senha, então será realizada uma verificação, se o endereço de e-mail e a senha forem válidos, se não forem válidos, o usuário irá para a tela de login caso contrário será direcionado para a tela inicial do aplicativo.

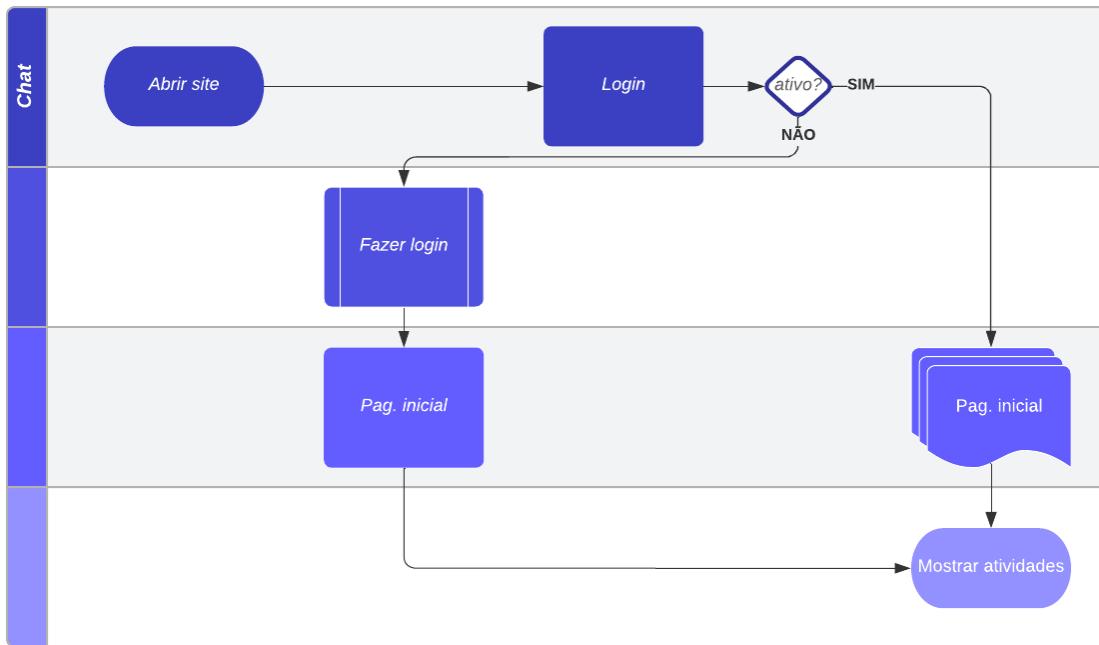
Caso o usuário não esteja cadastrado, o usuário será direcionado para a tela de cadastro de usuário onde será enviado um formulário para o usuário preencher os dados onde será inserido o nome, e-mail e senha. As informações de e-mail e senha devem ser verificadas, caso o endereço de e-mail ou senha não seja válido, o usuário será direcionado para a tela de cadastro do usuário, caso

contrário o cadastro será realizado e o usuário será direcionado para a tela de login.

3.7 Diagramas Estado

Um diagrama de estado, às vezes chamado de diagrama de máquina de estado, é um tipo de diagrama de comportamento na Unified Modeling Language (UML) que mostra transições entre diferentes objetos.

Figura 3.7: Diagrama de estado de login



Ao abrir o sistema ele verificará primeiro se o usuário está logado, se estiver o usuário será levado para a tela inicial, caso contrário para a tela de login onde será verificado se o usuário já está cadastrado, caso contrário o usuário será levado para a tela de cadastro do usuário, caso contrário, ele será direcionado para a tela de login. Se o usuário estiver cadastrado, o usuário fornecerá o e-mail e a senha, então será feita uma verificação, se o e-mail e a senha forem válidos, caso não sejam válidos, o usuário será levado para a tela de login. Caso o usuário não esteja cadastrado, o usuário será direcionado para a tela de cadastro de usuário onde será enviado um formulário para que o usuário preencha os dados onde estão inseridos nome, e-mail, telefone e senha.

As informações de e-mail, telefone e senha devem ser verificadas. Caso o endereço de e-mail, telefone ou senha não sejam válidos, o usuário será direcionado para a tela de cadastro do usuário, caso contrário o cadastro será realizado e o usuário será direcionado para a tela de login.



4. Projeto do Sistema OO

Este capítulo apresenta a arquitetura do sistema de classes, as interfaces do sistema e as tabelas de dados.

- Arquitetura do Sistema que será apresentada neste capítulo mostrará o diagrama de componente do sistema completo e de um subsistema específico.
- Capturas de tela de todas as interfaces de usuário do sistema
- Tabelas de dados que foram utilizadas no sistema.

Um projeto orientado a objetos é um processo de criação de um conjunto de modelos de projeto orientados a objetos que são posteriormente usados por programadores para escrever e testar o novo sistema a ser desenvolvido [SJB12].

4.1 Arquitetura do Sistema - Classes

- Arquitetura do sistema completo (Diagrama de Componentes UML)
- Arquitetura de um subsistema
- Arquitetura de

4.2 Interfaces do Usuário

Toda a UI foi desenvolvida pelo autor do trabalho usando Bootstrap, um framework CSS gratuito e de código aberto voltado para desenvolvimento web front-end responsivo e mobile-first. Inclui modelos de design baseados em HTML, CSS e JavaScript para tipografia, formulários, botões, navegação e outros componentes de interface.

- **Cadastro e Login do Usuário:** Um dos requisitos definidos foi que o usuário deve estar cadastrado e logado para acessar o sistema.

Sign up

Email

Password (*6 characters minimum*)

Password confirmation

[Sign up](#)

[Log in](#)

Figura 4.1: Tela de cadastro de usuário

Log in

Email

Password

Remember me

Log in

[Sign up](#)

[Forgot your password?](#)

Figura 4.2: Tela de login

- **Recuperação de senha:** Caso o usuário não consiga acessar o sistema será possível solicitar recuperação da senha cadastrada, encontrará a seguinte tela, indicando que o mesmo deve fazer para prosseguir.

Forgot your password?

Email

[Log in](#)

[Sign up](#)

Figura 4.3: Tela de recuperação de senha

- **Home:** Após o login, a página inicial é exibida primeiro. Sua função é apresentar algumas informações relevantes ao usuário.

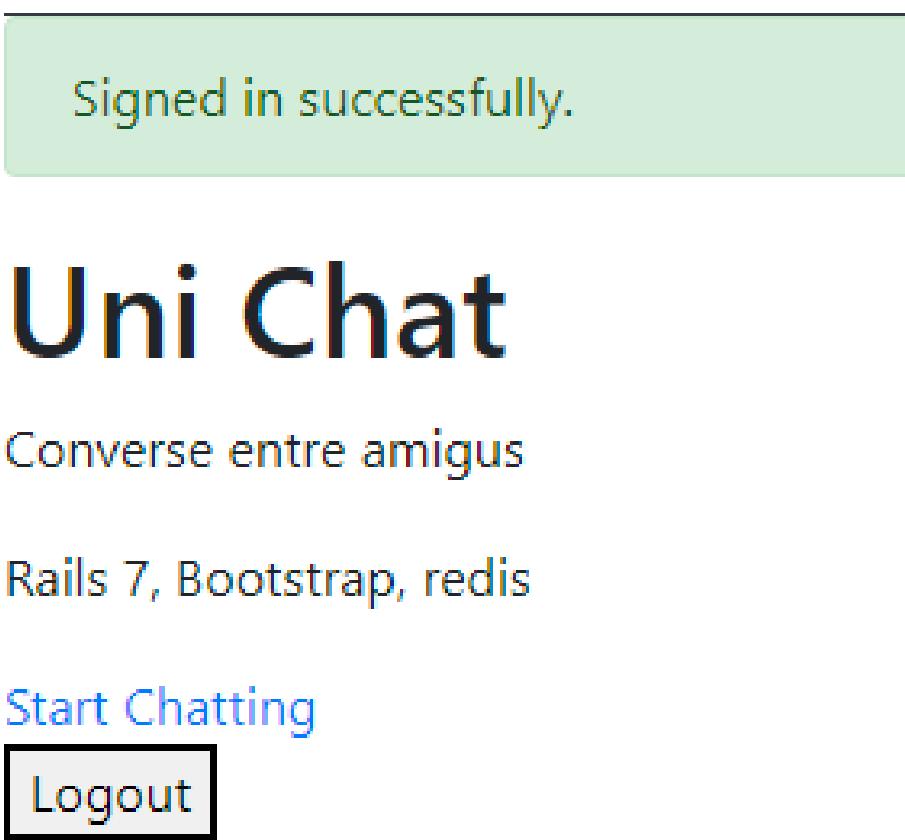


Figura 4.4: Tela inicial do sistema (home)

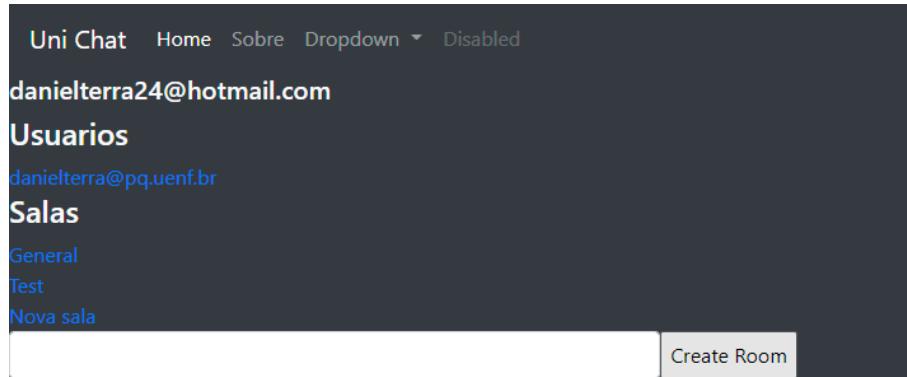


Figura 4.5: Tela inicial (chat)

4.3 Tabelas de Dados

Estruturas de dados que fazem parte da base de dados: cada uma com seus atributos e chaves principais e secundárias.

```

tic void Main(string[] args)
{
    for (int i = 0; i < 50; i++)
    {
        Thread mythread = new Thread(new Task());
        mythread.Start();
    }
}

Task.Run(() =>
{
    Console.WriteLine("starting task in thread " + Thread.CurrentThread.ManagedThreadId);
}

```



5. Implementação do Sistema OO

Este capítulo apresentará a implementação do sistema como as classes que foram desenvolvidas, os módulos que foram utilizados e a documentação do sistema com instruções para o usuário instalar e utilizar o sistema.

5.1 Programação

Para a construção do sistema foram utilizadas 3 tecnologias principais:

- Framework Ruby on Rails: criação da API (Interface de Programação de Aplicações) que disponibiliza o acesso aos dados.
- Ruby: Ruby é uma linguagem de programação interpretada, de alto nível e de propósito geral que suporta vários paradigmas de programação.
- SQLite: Biblioteca desenvolvida em C, que funciona como um banco de dados relacional.

5.1.1 Classes

Na modelagem do sistema desenvolvido foram definidas mais classes, no entanto, devido à complexidade e a fim de atender aos requisitos do trabalho proposto, somente foram implementadas as apresentadas a seguir:

```

1 class ConsultationsController < ApplicationController
2   before_action :set_consultation, only: [:show, :edit, :update, :destroy]
3   #### new line below
4   before_action :authenticate_user!, except: [:show, :index]
5   # GET /consultations or /consultations.json
6   def index
7     @consultations = Consultation.all
8   end
9
10  # GET /consultations/1 or /consultations/1.json
11  def show
12  end

```

```
13 # GET /consultations/new
14 def new
15   @consultation = Consultation.new
16 end
17
18
19 # GET /consultations/1/edit
20 def edit
21 end
22
23 # POST /consultations or /consultations.json
24 def create
25   @consultation = Consultation.new(consultation_params)
26   ## new line
27   # @consultation.user = current_user
28
29   respond_to do |format|
30     if @consultation.save
31       format.html { redirect_to consultation_url(@consultation), notice:
32         "Consultation was successfully created." }
33       format.json { render :show, status: :created, location:
34         @consultation }
35     else
36       format.html { render :new, status: :unprocessable_entity }
37       format.json { render json: @consultation.errors, status: :
38         unprocessable_entity }
39     end
40   end
41 end
42
43 # PATCH/PUT /consultations/1 or /consultations/1.json
44 def update
45   respond_to do |format|
46     if @consultation.update(consultation_params)
47       format.html { redirect_to consultation_url(@consultation), notice:
48         "Consultation was successfully updated." }
49       format.json { render :show, status: :ok, location: @consultation }
50     else
51       format.html { render :edit, status: :unprocessable_entity }
52       format.json { render json: @consultation.errors, status: :
53         unprocessable_entity }
54     end
55   end
56 end
57
58 # DELETE /consultations/1 or /consultations/1.json
59 def destroy
60   @consultation.destroy
61
62   respond_to do |format|
63     format.html { redirect_to consultations_url, notice: "Consultation
64     was successfully destroyed." }
65     format.json { head :no_content }
66   end
67 end
68
69 private
70   # Use callbacks to share common setup or constraints between actions.
71   def set_consultation
72     @consultation = Consultation.find(params[:id])
73   end
```

```

68 # Only allow a list of trusted parameters through.
69 def consultation_params
70   params.require(:consultation).permit(:title, :description, :
71   start_time, :end_time)
72 end
73 end

```

Listing 5.1: Ruby Controller

```

1 class LandsController < ApplicationController
2 def landpage
3   @consultations = Consultation.where(
4     start_time: Time.now.beginning_of_month.beginning_of_week..Time.now.
5     end_of_month.end_of_week
6   )
7 end
7 end

```

Listing 5.2: Ruby LandsController

5.1.2 Código Fonte

Aqui, serão apresentados alguns blocos de código extremamente importantes para o funcionamento do software. Após o login no sistema, a primeira página com a qual o usuário interage é a tela inicial. Esta página é composta por vários componentes e todos eles são "chamados" no código abaixo:

```

1 <div class="container mt-5 text-center">
2   <h1>Kalender</h1>
3   <p>Seja Bem-vindu:</p>
4
5   <h5> <%= current_user.email.split('@')[0].capitalize %> </h5>
6
7
8
9 <form action="http://[::1]:3000/consultations/new">
10 <button class="button1 button1">Criar nova atividade!</button>
11 </form>
12
13 <%= month_calendar(events: @consultations) do |date, consultations| %>
14   <div class="day">
15     <%= date %>
16   </div>
17   <% consultations.each do |consultation| %>
18     <div class="card-header">
19       <h5 class="card-title">
20         <%= link_to consultation.title, consultation %>
21       </h5>
22     </div>
23     <div class="card-body">
24       <p class="card-text">
25         <%= consultation.description %>
26       </p>
27     </div>
28     <div class="card-footer">
29       <p class="card-text">
30         From: <%= consultation.start_time.strftime('%A, %B %d, %Y at %I:%
M %p') %>

```

```

31      </p>
32      <p class="card-text">
33          To: <%= consultation.end_time.strftime('%A, %B %d, %Y at %I:%M %p
34          ') %>
35      </p>
36      </div>
37      <% end %>
38  <% end %>
</div>
```

Listing 5.3: Landpage

5.1.3 Banco da Dados

```

1 class CreateConsultations < ActiveRecord::Migration[7.0]
2   def change
3     create_table :consultations do |t|
4       t.string :title
5       t.text :description
6       t.datetime :start_time
7       t.datetime :end_time
8
9       t.timestamps
10    end
11  end
12 end
13 end
```

Listing 5.4: CreateConsultations

```

1 class AddAdminToUser < ActiveRecord::Migration[7.0]
2   def change
3     add_column :users, :admin, :boolean, default: false
4   end
5 end
```

Listing 5.5: AddAdminToUser

```

1 # frozen_string_literal: true
2
3 class DeviseCreateUsers < ActiveRecord::Migration[7.0]
4   def change
5     create_table :users do |t|
6       ## Database authenticatable
7       t.string :email, null: false, default: ""
8       t.string :encrypted_password, null: false, default: ""
9
10      ## Recoverable
11      t.string :reset_password_token
12      t.datetime :reset_password_sent_at
13
14      ## Rememberable
15      t.datetime :remember_created_at
16
17      ## Trackable
18      # t.integer :sign_in_count, default: 0, null: false
19      # t.datetime :current_sign_in_at
20      # t.datetime :last_sign_in_at
21      # t.string :current_sign_in_ip
```

```
23     # t.string      :last_sign_in_ip
24
25     ## Confirmable
26     # t.string      :confirmation_token
27     # t.datetime   :confirmed_at
28     # t.datetime   :confirmation_sent_at
29     # t.string      :unconfirmed_email # Only if using reconfirmable
30
31     ## Lockable
32     # t.integer    :failed_attempts, default: 0, null: false # Only if lock
33     strategy is :failed_attempts
34     # t.string      :unlock_token # Only if unlock strategy is :email or :both
35     # t.datetime   :locked_at
36
37     t.timestamps null: false
38 end
39
40 add_index :users, :email,                      unique: true
41 add_index :users, :reset_password_token, unique: true
42 # add_index :users, :confirmation_token,   unique: true
43 # add_index :users, :unlock_token,         unique: true
44 end
45 end
```

Listing 5.6: DeviseCreateUsers

5.1.4 Código Fonte - About Us

```
1 <div class="about-section">
2   <h1>Kalendar</h1>
3
4   <p class="div">Nossa missao e ser um calendario online que permite que um
5     ou mais usuarios editem e, opcionalmente, compartilhem com outros
6     usuarios o acesso online a um calendario.
7
8 A partir dos nossos servicos podera fazer uso de uma ferramenta
9     indispensavel. Pois, nossas vidas diarias sao gerenciaveis apenas
10    gracias aos nossos calendarios em nossos computadores e telefones. Um
11    calendario online permite-nos gerir os nossos horarios e ajuda-nos na
12    gestao do tempo. Pense em todas as vezes que as pessoas se esquecem de
13    suas reunioes e compromissos agendados. As consequencias incluem
14    remarcar as reunioes para uma data muito posterior ou ate mesmo pagar
15    uma taxa por faltar a um compromisso. Um calendario online permite que
16    voce sempre acompanhe suas reunioes e compromissos.</p>
17
18 </div>
19
20 <h2 style="text-align:center">Nosso Time</h2>
21 <div class="row">
22   <div class="column">
23     <div class="card">
24       
28       <div class="container">
29         <h2>Daniel Terra Gomes</h2>
```

```

17      <p class="title">Estudante</p>
18      <p>Computer Science | Machine Learning Enthusiast | AI Ethics</p>
19      <p>danielterra@pq.uenf.br</p>
20      <p><button class="button">Contato</button></p>
21      </div>
22  </div>
23</div>
24
25<div class="column">
26  <div class="card">
27    
28    <div class="container">
29      <h2>Ausberto S. Castro Vera</h2>
30      <p class="title">Professor</p>
31      <p>PhD Computer Science: Software Engineering, Requirements
Engineering, Computer Security, Cloud Computing</p>
32      <p>ascv@uenf.br</p>
33      <p><button class="button">Contato</button></p>
34    </div>
35  </div>
36</div>
37
38</div>
39

```

Listing 5.7: About Us

5.2 Documentação do Software

Manual de Instalação

- **Manual de Instalação:** O que é necessário para instalar o sistema desenvolvido (banco de dados, versão, bibliotecas, etc.)
- **Manual do usuário:** Os passos básicos para utilizar o sistema (inicializar, salvar, imprimir, etc.)
- **Outros....**

Software Version

- Ruby version: ruby-3.1.2
- SQLite version: 1.4

Configuration

- delete Gemfile.lock
- update Ruby version of Gemfile to your version
- then Bundle update
- bundle install

Database initialization: rails db:migrate

Seguindo essas etapas, agora você deve conseguir acessar o sistema no localhost. Quaisquer dúvidas relacionadas ao funcionamento do sistema podem ser esclarecidas na seção Interface do Usuário.



6. Considerações Finais

O trabalho desenvolvido se trata de um sistema que permite ao usuário criar atividades em uma agenda e compartilhar com seus colegas de empresa. Este trabalho teve como objetivo desenvolver um sistema que aplique os conhecimentos adquiridos no curso para obtenção de nota de crédito na disciplina de paradigma orientado a objetos para desenvolvimento de software, além de recordar os conceitos das disciplinas do curso de Ciência da Computação. Entre os problemas encontrados na realização do trabalho, o mais importante foi o desenvolvimento da parte escrita. No entanto, essas dificuldades contribuem para o desenvolvimento de soft skills e competências técnicas necessárias no mercado de trabalho, tais como: Compreensão da demanda de trabalho, priorização, gestão do tempo. Os aspectos não considerados úteis para o sistema seriam: o desenvolvimento de uma interface para o cliente onde pudesse haver uma criação de atividade para que o gerenciamento das tarefas dos funcionários da empresa; configurar uma página exclusiva para pessoas de fora da empresa.

Recursos a serem desenvolvidos no futuro:

- Troca de mensagens entre o usuário e o criador da atividade na agenda;
- Favoritos: O sistema permitirá que os usuários possam marcar as propriedades que deseja adicionar aos favoritos.
- Editar informações do usuário: bem como recurso favorito, o sistema possuirá personalização para que o usuário possa modificar as informações do usuário.



Figura 6.1: Meu Sistema a ser desenvolvido



Referências Bibliográficas

- [dev22] devashishakula503. What is a distributed system?, May 2022. Citado na página 1.
- [DWR14] Alan Dennis, Barbara Haley Wixom, and Roberta M. Roth. *Análise e Projeto de Sistemas*. LTC, Rio de Janeiro, 5 edition, 2014. Citado 2 vezes nas páginas 1 e 6.
- [Fur13] Sérgio Furgeri. *Modelagem de Sistemas Orientados a Objetos*. Érica Editora, São Paulo, SP, 1 edition, 2013. Citado na página 1.
- [Gue11] Gilleanes T.A. Guedes. *UML 2 : uma abordagem prática*. Novatec Editora, 2011. Citado na página 1.
- [Hel13] Helio Engholm Jr. *Análise e Design Orientados a Objetos*. Novatec, 2013. Citado na página 1.
- [SJB12] John W. Satzinger, Robert B. Jackson, and Stephen D. Burd. *Introduction to Systems Analysis and Design: An Agile, Iterative Approach*. Course Technology, CENGAGE Learning, Mason, Ohio, 6 edition, 2012. Citado 2 vezes nas páginas 1 e 17.
- [Som18] Ian Sommerville. *Engenharia de Software*. Pearson Education do Brasil, São Paulo, 10 edition, 2018. Citado na página 1.
- [SR12] Gary B. Shelly and Harry J. Rosenblat. *Analysis and Design for Systems*. Course Technology, CENGAGE Learning, 9 edition, 2012. Citado na página 1.
- [Waz11] Raul Sidnei Wazlawick. *Análise e Projeto de Sistemas de Informação Orientados a Objetos*. Editora Campus SBC. Elsevier, Rio de Janeiro, RJ, 2 edition, 2011. Citado 2 vezes nas páginas 1 e 5.