**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**

**FATEC DE PRAIA GRANDE**

**Desenvolvimento de Software Multiplataforma**

**Ana Beatriz Fernandes Caldeira da Silva**

**Bruna Martins Santana**

**Julia Lopes Ribeiro**

**Maria Eduarda Fontes dos Santos**

**FATECCONNECT**

**Praia Grande – SP**  
**2025**

**Ana Beatriz Fernandes Caldeira da Silva**

**Bruna Martins Santana**

**Julia Lopes Ribeiro**

**Maria Eduarda Fontes dos Santos**

**FATECCONNECT**

Projeto Interdisciplinar apresentado na disciplina de Laboratório de Desenvolvimento Web como requisito para a conclusão do 4º Módulo do Curso de Técnologia de Desenvolvimento de Software Multiplataforma.  
Orientadora: Profª Eulaliane A. Gonçalves

**Praia Grande – SP**

**2025**

**SUMÁRIO**

[1. INTRODUÇÃO 3](#_Toc608525676)

[1.1. Tema 3](#_Toc1264442343)

[1.2. Objetivos 4](#_Toc792410949)

[1.3. Problematização 4](#_Toc1859383685)

[1.4. Justificativa 5](#_Toc1677730460)

[1.5. Metodologia 5](#_Toc1513054289)

[1.6. Organização do Trabalho 5](#_Toc1692356275)

[2. DESCRIÇÃO GERAL DO SISTEMA 6](#_Toc1473013498)

[2.1 Descrição do Problema 6](#_Toc455017608)

[2.2 Principais Envolvidos e suas Características 7](#_Toc531570930)

[2.3 Regras de Negócio 8](#_Toc178399815)

[3. REQUISITOS DO SISTEMA 9](#_Toc1552368465)

[3.1 Requisitos Funcionais 9](#_Toc1500892972)

[3.2 Requisitos Não Funcionais 10](#_Toc1830540984)

[3.3 Protótipos de Tela 10](#_Toc732006738)

[3.3.1 Diagrama de Navegação 22](#_Toc1704407122)

[4. ARQUITETURA DO SISTEMA 23](#_Toc2083066821)

[4.1. Componentes Principais 24](#_Toc830252599)

[• Front-End: A interface do usuário foi desenvolvida com a biblioteca React.js, proporcionando uma experiência interativa e responsiva. 24](#_Toc1813524930)

[• Back-End: A API RESTful foi desenvolvida utilizando Node.js e o framework Express, responsável por gerenciar toda a lógica de negócio, autenticação e integração com o banco de dados. 24](#_Toc1011219768)

[4.2. Diagrama de Arquitetura 24](#_Toc1489774757)

[4.3. Segurança 25](#_Toc601718016)

[4.4. Escalabilidade 25](#_Toc22188427)

[4.5. Manutenção 25](#_Toc654939185)

[ Estrutura modular com separação entre rotas e controladores. 25](#_Toc1845662948)

[4.6. Integração 25](#_Toc1216753918)

[4.7. Limitações 26](#_Toc1636147006)

[5. ARQUITETURA DE DADOS 26](#_Toc1043346507)

[5.1. Modelo Lógico da Base de Dados 26](#_Toc205743534)

[5.2. Criação Física do Modelo de Dados 27](#_Toc1037699696)

[5.3. Dicionário de Dados 28](#_Toc1128838091)

[6. PROJETO DE SOFTWARE 29](#_Toc2069137663)

[6.1. Design dos Componentes 29](#_Toc85850028)

[6.2. Diagrama de Classes 31](#_Toc1848517868)

[6.3. Diagrama de Sequência ou Fluxo de Interações 32](#_Toc602024966)

[6.4. Regras de Negócio Detalhadas 33](#_Toc265565087)

[6.5. Estratégias de Tratamento de Erros 35](#_Toc2071495852)

[7. IMPLEMENTAÇÃO 36](#_Toc1931965054)

[7.1. Estrutura geral do projeto 36](#_Toc1906852060)

[7.2. Modelos 37](#_Toc1554371858)

[7.2.1 Modelo User 37](#_Toc1527259379)

[7.2.2 Modelo Curso 38](#_Toc1962397496)

[7.2.3 Modelo HorarioDisponivel 38](#_Toc617604218)

[7.2.4 Modelo Agendamento 39](#_Toc841824914)

[7.3. Views (Routes) 39](#_Toc1998670531)

[7.3.1. View para Criar Agendamento 39](#_Toc850587369)

[7.4 Templates (Componentes) 40](#_Toc713243352)

[7.4.1 Template para Exibir os Agendamentos 40](#_Toc623746123)

# 1. INTRODUÇÃO

Esse trabalho tem como objetivo desenvolver uma plataforma digital chamada FatecConnect, destinada a facilitar a interação entre monitores e alunos da FATEC. A proposta surge da necessidade de modernizar o sistema atual de monitoria, que, apesar de funcional, apresenta limitações que dificultam o acesso eficiente ao suporte acadêmico que é oferecido pela instituição. A FatecConnect visa oferecer uma solução prática e acessível, com funcionalidades como chat em tempo real, fóruns de discussões e um sistema de agendamento de monitorias presenciais. Além disso, a plataforma incluirá uma seção dedicada à divulgação de vagas, bootcamps, cursos gratuitos, que poderão ser incluídos pelos próprios alunos.

## 1.1. Tema

O projeto tem como objetivo criar um site onde monitores e alunos da FATEC possam interagir de forma prática e eficiente. A plataforma contará com funcionalidades como chat em tempo real, fóruns e um sistema de agendamento de monitorias presenciais, permitindo que os alunos tirem dúvidas relacionadas às suas respectivas matérias e cursos, desde questões simples até as mais complexas. A proposta é simplificar um processo que, para muitos alunos, pode ser complicado, como ter que alterar seu cronograma para esclarecer dúvidas aparentemente simples. No projeto, os monitores terão acesso às dúvidas publicadas pelos alunos, enquanto os estudantes poderão buscar um atendimento simplificado pelos monitores disponíveis em seu curso. Além disso, a plataforma incluirá um espaço para a divulgação de oportunidades de estágios.

## 1.2. Objetivos

Desenvolver um fórum educativo para alunos da FATEC, facilitando a interação com monitores e promovendo um ambiente organizado para perguntas e respostas acadêmicas.

O sistema tem como objetivos específicos a criação de um espaço interativo destinado a esclarecer dúvidas acadêmicas, com a moderação de monitores, para garantir que as informações sejam precisas e úteis. Além disso, permitirá o agendamento de monitorias individuais, oferecendo um suporte mais personalizado aos alunos que necessitam de auxílio em suas atividades acadêmicas.

Outra funcionalidade importante será a disponibilização de filtros por disciplina, facilitando a busca por respostas relacionadas a temas específicos, otimizando o tempo dos usuários ao procurarem informações relevantes. O sistema também enviará notificações para manter os usuários informados sobre atualizações relevantes, como respostas aprovadas ou agendamentos confirmados, garantindo que todas as partes envolvidas estejam sempre atualizadas sobre o andamento das interações.

Além disso, será incluída uma seção com oportunidades de estágio e emprego voltadas para os cursos da FATEC, oferecendo aos alunos acesso a informações sobre vagas que possam contribuir para o desenvolvimento de sua carreira profissional.

## 1.3. Problematização

Como alunas da FATEC, percebemos que o sistema tradicional utilizado atualmente pode ser considerado arcaico, tornando o processo de tirar dúvidas mais trabalhoso do que realmente deveria ser. Hoje, um aluno precisa agendar um horário para monitoria, deslocar-se até o local agendado e, só então, retirar a dúvida. Esse processo, muitas vezes, é mais complexo e demorado do que a dúvida em si. Como resultado, muitos alunos acabam recorrendo a plataformas digitais de inteligência artificial, já que, muitas vezes, a dúvida é simples e pode ser resolvida de forma mais rápida e prática no celular, ao invés de ir à monitoria presencial. Esse cenário evidencia a necessidade de uma solução mais eficiente e acessível para facilitar o aprendizado.

## 1.4. Justificativa

Para tal problema propusemos um software que visa incentivar alunos da FATEC a frequentarem a monitoria, visto que, grande parte das pessoas não usufruem da mesma. Com isso, a FatecConnect propõe uma forma mais acessível e prática de estudos, assim aproximando os alunos da monitoria.

1.5. Metodologia

Para o desenvolvimento do projeto, será adotada a metodologia Ágil, utilizando o framework SCRUM. O SCRUM organiza o trabalho em ciclos curtos chamados sprints, permitindo entregas frequentes, feedback contínuo e maior flexibilidade para ajustes ao longo do processo. Com essa abordagem, a equipe poderá desenvolver o sistema de forma colaborativa e iterativa, garantindo que o produto atenda às necessidades dos usuários.

O projeto será estruturado com base no paradigma orientado a objetos (OO), visando modularidade, reutilização de código e melhor organização da aplicação. A implementação utilizará React.js no frontend e Node.js no backend, proporcionando uma solução moderna e escalável.

O banco de dados escolhido será o PostgreSQL, garantindo armazenamento eficiente das perguntas e respostas. Além disso, o sistema contará com WebSockets, possibilitando comunicação em tempo real entre alunos e monitores. Para validar a usabilidade e aderência do sistema ao público-alvo, será aplicada uma pesquisa com os alunos da FATEC, assegurando que o desenvolvimento atenda às suas necessidades.

## 1.6. Organização do Trabalho

A introdução abrange o tema do projeto, seus objetivos gerais e específicos, a problematização destacando as dificuldades enfrentadas pelos alunos da FATEC no modelo atual de monitoria, e a justificativa para a criação da plataforma. Além disso, apresenta a metodologia adotada, que inclui o uso da metodologia Ágil, o paradigma de orientação a objetos e as tecnologias empregadas no desenvolvimento do sistema.

# 2. DESCRIÇÃO GERAL DO SISTEMA

Este capítulo apresenta uma visão geral do sistema proposto, abordando os principais problemas que ele busca resolver, os envolvidos no processo e as regras de negócio que garantem seu funcionamento eficiente. A plataforma FatecConnect foi desenvolvida para modernizar o serviço de monitoria da FATEC, oferecendo uma solução digital que facilita a interação entre alunos e monitores, promove a organização do suporte acadêmico e incentiva a participação dos estudantes. A seguir, são detalhados os aspectos essenciais do sistema, incluindo a descrição do problema, os perfis dos usuários envolvidos e as diretrizes que regem sua operação.

## 2.1 Descrição do Problema

O sistema em questão é direcionado à faculdade FATEC e visa melhorar o serviço de monitoria oferecido aos alunos. Atualmente, a monitoria enfrenta desafios relacionados à pouca divulgação, o que pode resultar em uma baixa adesão por parte dos alunos que necessitam desse suporte. Além disso, a organização do processo de monitoria também é um ponto de melhoria.

O sistema afeta diretamente a forma atual de monitoria da faculdade, que sofre com a falta de divulgação, podendo não alcançar todos os alunos que necessitam desse apoio. Muitos alunos podem não estar cientes da existência do serviço de monitoria ou não sabem como acessá-lo de forma eficiente.

O principal impacto esperado do sistema é a melhoria da interação entre alunos e monitores, tornando o acesso ao serviço de monitoria mais simples e organizado. A utilização do sistema visa aumentar a visibilidade das monitorias e incentivar os estudantes a utilizarem esse recurso, promovendo um ambiente acadêmico mais colaborativo.

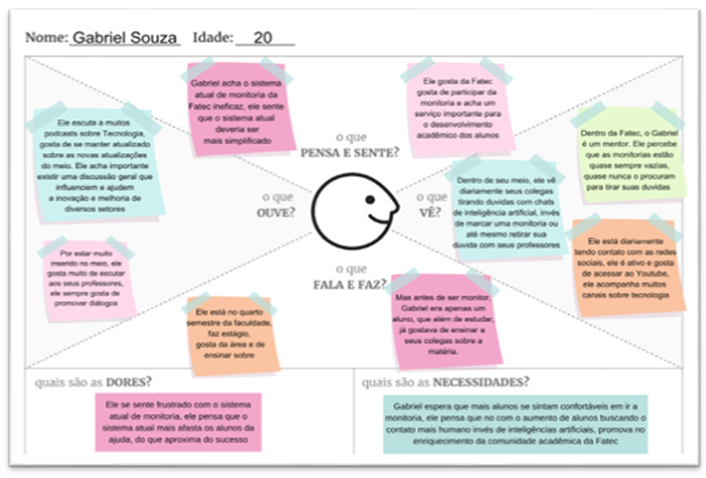
Uma solução eficaz seria aumentar a divulgação do serviço de monitoria de forma estratégica, otimizar o tempo de alunos e monitores através de agendamentos eficientes, e criar um fórum de perguntas e respostas onde as dúvidas possam ser tiradas de maneira mais ágil e dinâmica, facilitando a comunicação entre todos os envolvidos.

## 2.2 Principais Envolvidos e suas Características

O sistema proposto destina-se aos alunos da FATEC, sendo desenvolvido com foco em dois perfis principais de usuários: alunos e alunos monitores. Seu principal objetivo é proporcionar um ambiente virtual que facilite o agendamento de sessões de monitoria, o acesso a materiais de apoio didático, além de promover uma interação mais eficiente por meio de um fórum de dúvidas.

Os alunos poderão consultar os horários de monitoria disponíveis, realizar agendamentos, acessar conteúdos complementares e interagir no fórum com o envio de perguntas relacionadas às disciplinas. Os alunos monitores, além de utilizarem essas funcionalidades, contarão com permissões adicionais no sistema, como o gerenciamento de suas sessões de monitoria, a moderação de postagens e a resposta prioritária às dúvidas, garantindo a organização da interação na plataforma. Proporcionando um ambiente intuitivo, acessível e funcional que atenda às necessidades de ambos os lados.

* Mapa da Empatia

  
Figura 1 – Mapa da Empatia.

O sistema está sendo desenvolvido por uma equipe composta por quatro discentes, sendo duas responsáveis pelo desenvolvimento do backend e duas pelo desenvolvimento do frontend. As desenvolvedoras de backend, Ana Beatriz Fernandes Caldeira da Silva e Maria Eduarda Fontes dos Santos, são responsáveis pela construção das funcionalidades internas do sistema, incluindo a lógica de funcionamento, a integração com o banco de dados e a implementação das regras de negócio, assegurando a correta operação do sistema.

As desenvolvedoras de frontend, Bruna Martins Santana e Julia Lopes Ribeiro, são responsáveis pela elaboração da interface gráfica, buscando proporcionar uma experiência de uso intuitiva, acessível e visualmente agradável aos usuários. Suas atividades envolvem a estilização das páginas e a integração com as funcionalidades desenvolvidas no backend.

O trabalho em equipe entre o backend e frontend é crucial para garantir qualidade e eficiência no desenvolvimento do sistema, assim assegurando a entrega de um produto funcional, confiável e adaptado às necessidades dos alunos da FATEC.

## 2.3 Regras de Negócio

O Fórum de Monitoria da FATEC tem como principal objetivo facilitar a interação entre alunos e monitores, criando um ambiente colaborativo propício para a troca de conhecimentos. Para garantir o bom funcionamento desse sistema, foram estabelecidas algumas regras e diretrizes.

Primeiramente, o acesso ao fórum será restrito apenas a alunos e monitores que estejam devidamente cadastrados no sistema. Além disso, os monitores terão permissões adicionais para gerenciar postagens e responder dúvidas de forma prioritária. A fim de preservar um ambiente respeitoso, não será permitida a utilização de linguagem inadequada ou ofensiva, sendo que todas as postagens e respostas devem estar relacionadas ao conteúdo acadêmico, evitando spam ou discussões irrelevantes.

No que diz respeito ao desempenho, o sistema deve ser capaz de suportar um número elevado de acessos simultâneos sem comprometer sua performance. As mensagens do chat em tempo real devem ser entregues com baixa latência, garantindo fluidez na comunicação. O tempo de resposta para consultas de postagens deve ser inferior a 2 segundos, mesmo com grande volume de dados sendo processados.

Em termos de tolerância a falhas, o sistema precisa garantir a persistência das mensagens e postagens, evitando qualquer tipo de perda de informações em caso de falhas inesperadas. Caso ocorra falha na conexão, o usuário deve ser notificado e a tentativa de reconexão deve ser realizada automaticamente. Além disso, o fórum deve ser acessível de diferentes dispositivos, incluindo desktops e dispositivos móveis.

Com relação ao volume de informações, o sistema deve ser capaz de armazenar um histórico de postagens e interações, garantindo que essas informações estejam acessíveis futuramente. Estima-se que o número de usuários ativos crescerá cerca de 10% ao mês, e o banco de dados deve ser escalável para acompanhar o crescimento tanto da base de dados quanto do número de interações.

Por fim, as ferramentas de apoio que serão utilizadas no sistema incluem o PostgreSQL para o banco de dados, garantindo robustez e escalabilidade; o Node.js com Express para o backend, facilitando a comunicação entre o frontend e o banco de dados; o React.js no frontend, proporcionando uma experiência interativa e responsiva; WebSockets com Socket.io para o sistema de chat em tempo real; e servidores na nuvem para garantir alta disponibilidade no processo de hospedagem e deploy.

# 3. REQUISITOS DO SISTEMA

## 3.1 Requisitos Funcionais

Os requisitos funcionais são as funcionalidades e comportamentos específicos que o sistema deve ter para atender às necessidades do usuário descrevendo o que essas funcionalidades devem fazer.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Descrição | Prioridade | Regra de Negócio Relacionada |
| RF01 | Permitir cadastro e login de usuários (alunos/monitores) com validação de e-mail institucional (@fatec.sp.gov.br). | Alta | Seção 1.1 (Autenticação e Controle de Acesso) |
| RF02 | O sistema deve diferenciar perfis (aluno/monitor), concedendo a monitores permissões para moderar fóruns e gerenciar agendamentos. | Alta | Seção 2.2 (Autenticação e Controle de Acesso) |
| RF03 | Permitir que alunos postem dúvidas por disciplina. | Alta | Seção 2.2 (Fórum de Dúvidas) |
| RF04 | Deve exibir um fórum de dúvidas por disciplina (Ex: “Banco de Dados Relacional”, “Algoritmos e Lógica de Programação”). | Alta | Seção 2.2 (Fórum de Dúvidas) |
| RF05 | Oferecer um chat privado entre aluno e monitor. | Média | Seção 3.1 (Comunicação em Tempo Real) |
| RF06 | O sistema deve exibir os horários disponíveis dos monitores por disciplina. | Média | Seção 4.1 (Agendamento de Monitorias) |
| RF07 | Permitir agendamento de monitorias com confirmação por e-mail. | Média | Seção 4.2 (Agendamento de Monitorias) |
| RF08 | O sistema deve permitir que postem vagas de estágio. | Baixa | Seção 5.1 (Divulgação de Vagas) |
| RF09 | Notificar alunos quando novas vagas compatíveis com seus interesses forem postadas. | Baixa | Seção 5.2 (Divulgação de Vagas) |

## 3.2 Requisitos Não Funcionais

Os requisitos não funcionais servem para definir como o sistema deve operar, ao invés de descrever o que ele deve fazer. Estão relacionados à performance, segurança, usabilidade, entre outros aspectos qualitativos.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Categoria | Descrição | Métrica/Valor Esperado |
| RNF01 | Desempenho | O sistema deve suportar até 500 usuários simultâneos | 500 conexões simultâneas |
| RNF02 | Desempenho | O chat em tempo real deve ter baixa latência | Máximo 1 segundo de latência |
| RNF03 | Segurança | Dados devem ser criptografados e senhas armazenadas com hash | SHA-256 ou superior |
| RNF04 | Segurança | Controle de moderação no fórum | Apenas monitores podem moderar postagens |
| RNF05 | Usabilidade | Interface responsiva e acessível | Compatível com mobile/desktop |
| RNF06 | Disponibilidade | Atendimento com o monitor disponível em horário comercial | (7h-22h) |
| RNF07 | Compatibilidade | Funcionamento em navegadores modernos | Chrome, Firefox, Edge (últimas versões) |

## 3.3 Protótipos de Tela

Este capítulo apresenta os protótipos de alta fidelidade do site, desenvolvidos no Figma, com o objetivo de ilustrar a interface e a experiência do usuário nas principais telas do sistema.

* **Tela inicial do projeto:**

A tela inicial tem como objetivo apresentar o site do FatecConnect, destaca os principais recursos e incentiva o cadastro dos usuários. Ela pode ser acessada ao entrar no site.

Funcionalidades e regras:

* Logo da página que redireciona para a página inicial.
* Link “Inicio”: Redireciona para o início.
* Link “Contato”: Redireciona para a página de contatos.
* Botão “Login”: Redireciona para a tela de login.
* Botão “Cadastre-se”: Redireciona para a página de cadastro.

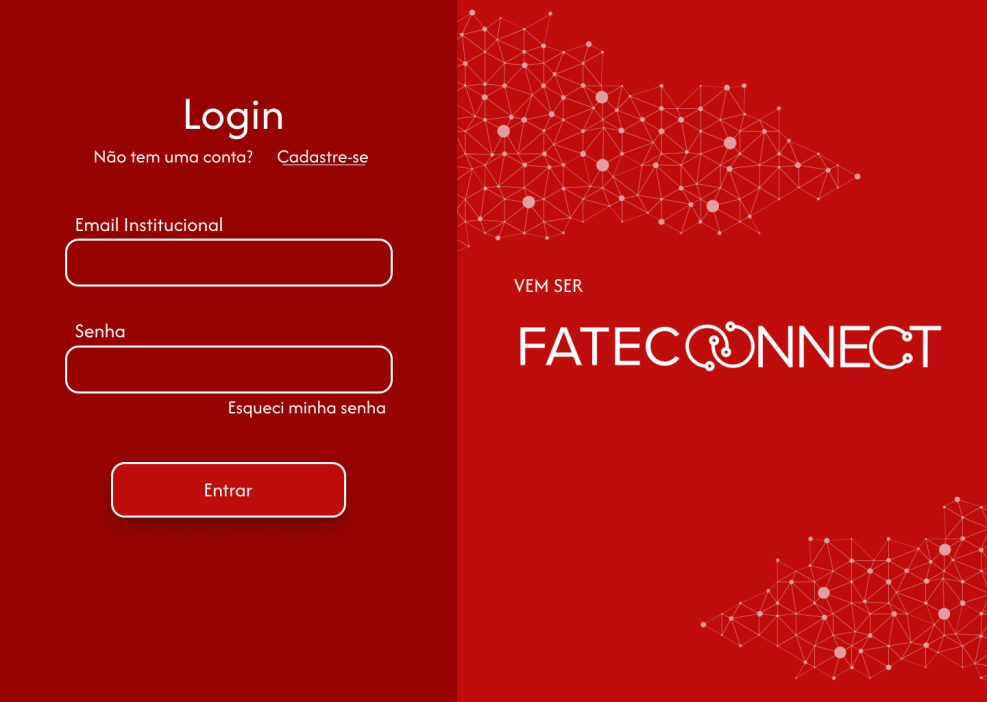
  
Figura 2 - Protótipo da Tela Inicial.

* **Tela de login:**

Permitir que usuários cadastrados acessem suas contas no FatecConnect. Pode ser acessada pelo botão "Login" no menu principal ou ao tentar acessar uma funcionalidade restrita sem estar autenticado.

Funcionalidades e regras:

* Link “Cadastre-se”: Redireciona para a página de cadastro.
* Campo “E-mail Institucional”: Campo de texto, obrigatório, até 50 caracteres, alfanumérico. Aceita apenas e-mails cadastrados e válidos (@fatec.sp.gov.br).
* Campo “Senha”: Campo de senha, obrigatório, até 20 caracteres, alfanumérico.
* Botão "Entrar": Realiza o login após validação dos campos.
* Link "Esqueci minha senha": Redireciona para a página de recuperação de acesso.
* Qualquer visitante pode visualizar a página, mas apenas usuários cadastrados podem fazer login.

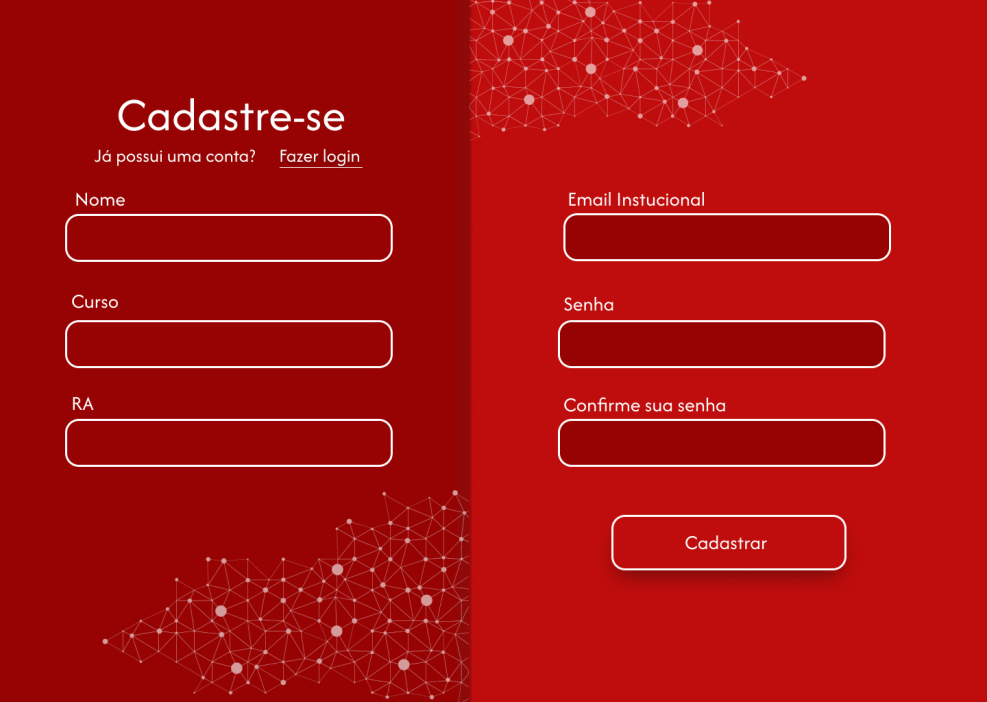
  
Figura 3 - Protótipo da Tela de Login.

* **Tela de cadastro:**

Permitir que novos usuários criem uma conta no FatecConnect.Pode ser acessada pelo botão "Cadastrar-se" na tela de login ou no menu principal.

Funcionalidades e regras:

* Campos obrigatórios: Nome, E-mail, Senha e Confirmação de Senha.
* O campo de e-mail deve aceitar apenas endereços válidos (@fatec.sp.gov.br).
* A senha deve ter no mínimo 6 caracteres e deve coincidir com a confirmação de senha.
* Botão "Cadastrar" para finalizar o registro.
* Link "Já possui uma conta? Fazer login" para usuários que já possuem cadastro
* Qualquer visitante pode visualizar e utilizar a tela para criar uma conta.

  
Figura 4 - Protótipo da Tela de Cadastro.

* **Tela de Redefinição de Senha**

O objetivo dessa página é permitir que o usuário possa redefinir sua senha. Pode ser acessada pela página de login no link “esqueci minha senha”.

Funcionalidades e regras:

* Campos obrigatórios: Nova senha e confirmar senha.
* Ambos os campos, deve ter no mínimo 6 caracteres, alfanuméricos e a confirmação deve coincidir com o a nova senha.
* Botão "Confirmar" para finalizar o registro.
* Qualquer visitante pode visualizar e utilizar a tela para alterar sua senha, desde que já tenha uma senha para alterar.



Figura 4 - Protótipo da Tela de Redefinição de Senha.

* **Painel do aluno**

O objetivo dessa página é permitir que o aluno gerencie suas atividades, assim como, alterar biografia, fazer novas publicações e ver publicações antigas. Para o aluno ter acesso ao seu perfil, anteriormente ele deve fazer seu cadastro na área de login e assim chegando ao seu painel.

Funcionalidades e regras:

* Barra de navegação: Possui links para o chat de mensagens.
* Botão "Logout": Permite que o aluno saia do perfil.
* Icone de câmera: Permite que a foto seja alterada.
* Campos de respostas permitem textos longos, sendo no máximo até 5000 caracteres.
* Campo “Posts”: Permite que o usuário faça novas publicações com imagens/vídeos.
* Todas os usuários cadastrados podem visualizar essa página.

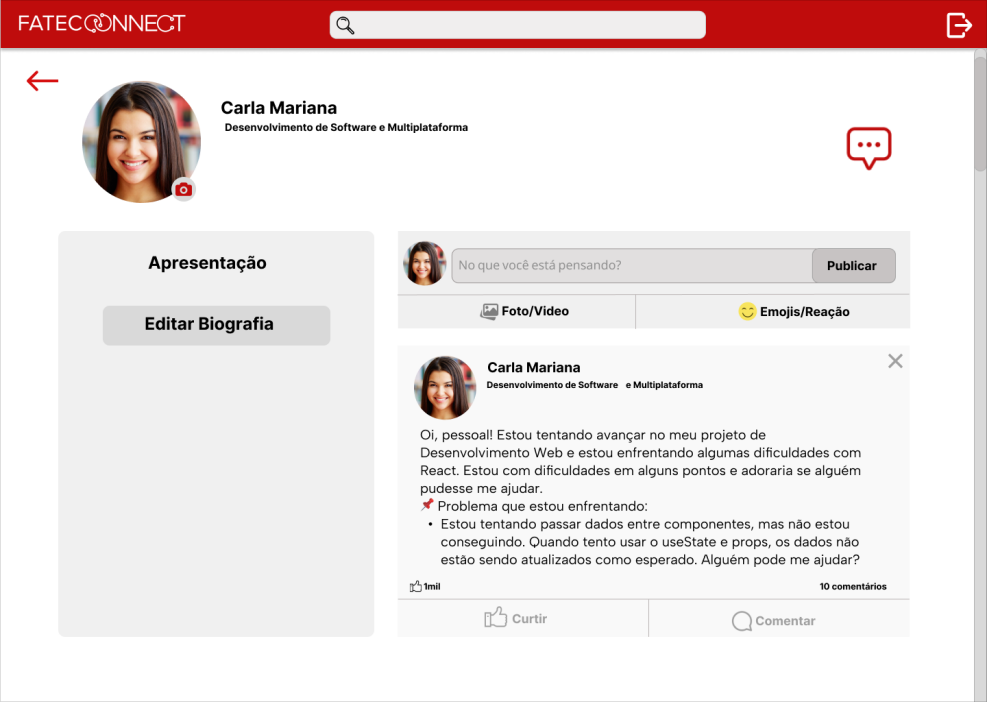


Figura 5 - Protótipo da Tela do Painel do Aluno.

* **Painel do monitor**

O objetivo dessa página é permitir que o monitor gerencie suas atividades, assim como, notificações dos agendamentos, mensagens e curtidas. Para o monitor ter acesso ao seu perfil, anteriormente ele deve ser cadastrado pelo admin como monitor, assim passando pelo login e chegando ao seu painel.

Funcionalidades e regras:

* Barra de navegação: Possui links para o chat de mensagens, agenda de monitoria e curtidas ou comentários dos seus posts.
* Botão "Logout": Permite que o monitor saia do perfil.
* Botão para editar foto: Atualiza a foto.
* Botão “Comentar”: Permite que o monitor responda as dúvidas dos estudantes.
* Campos de respostas permitem textos longos, sendo no máximo até 5000 caracteres.
* Apenas monitores cadastrados podem visualizar essa página.

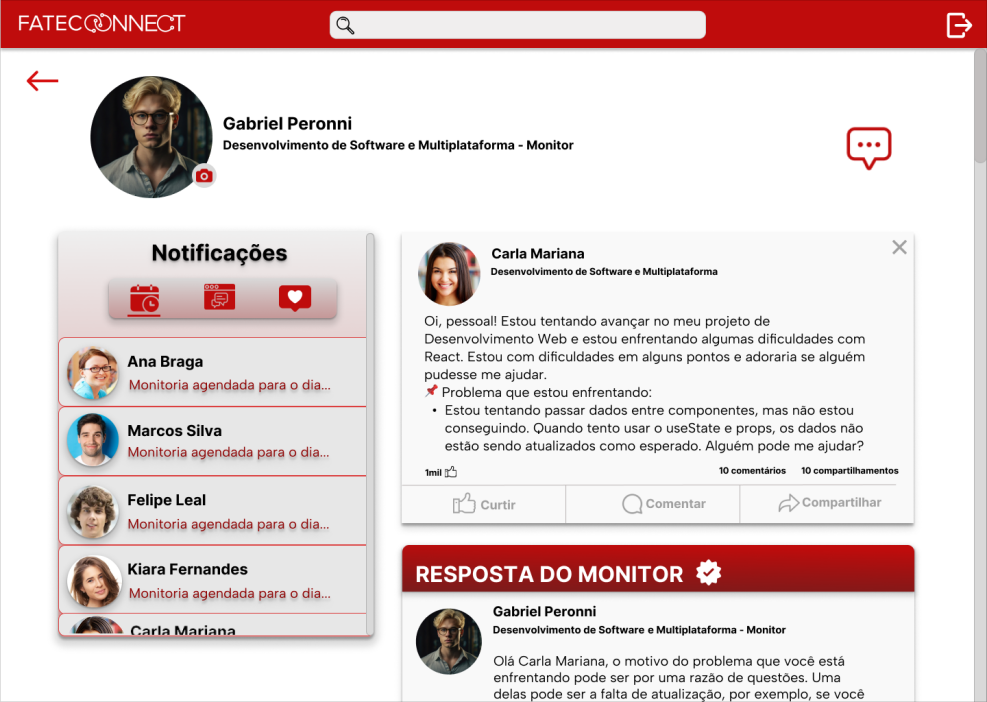


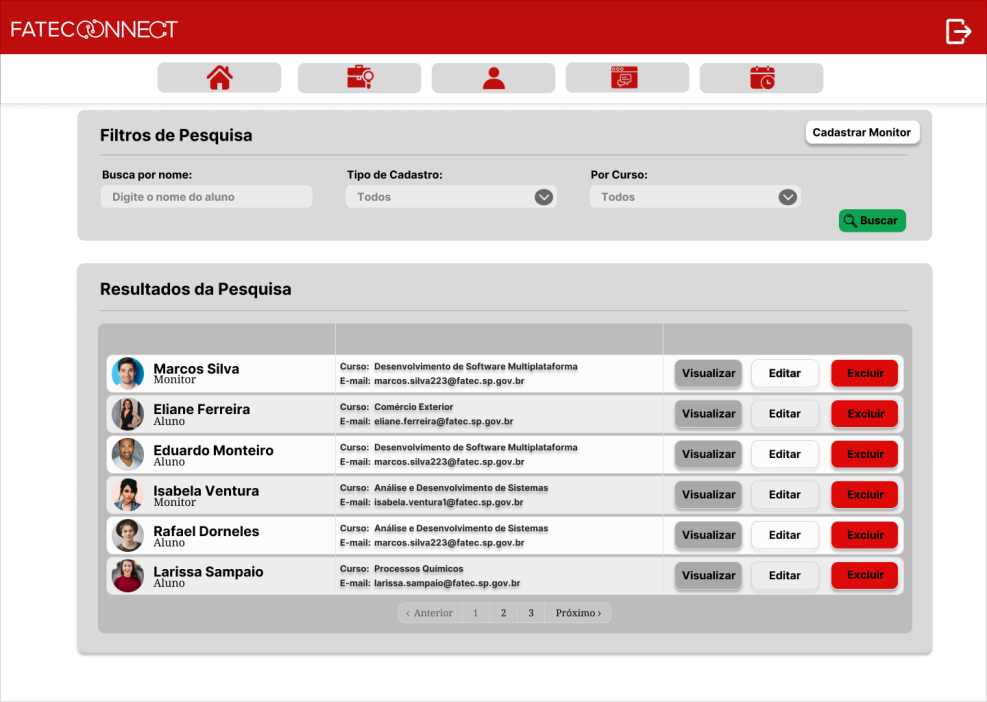
Figura 6 - Protótipo da Tela do Painel do Monitor.

* **Painel do admin**

Essa página é restrita e apenas o usuário admin tem acesso à suas funcionalidades, o objetivo é o controle de usuários, como a criação de novos usuários monitor, edição, visualização e exclusão de cadastros.

Funcionalidades e regras:

* Barra de navegação: Possui links para o chat de mensagens, agenda de monitoria e curtidas ou comentários dos seus posts.
* Botão "Logout": Permite que o monitor saia do perfil.
* Area destinada para pesquisa de usuários.
* Campo “Busca por nome”: Permite pesquisar por nome entre os cadastros.
* Campo “Tipo de Cadastro”: Permite selecionar um filtro por tipo do cadastro, com opções como “Monitor” e “Aluno”.
* Campo “Por Curso”: Permite selecionar o curso desejado para filtrar os cadastros.
* Botão “Cadastrar Monitor”: Redireciona para a página de cadastro de monitor, que apenas o usuário admin tem acesso.
* Botão “Buscar”: Botão que confirma os campos que foram selecionados para filtragem de pesquisa, podendo ser um, dois ou todos os tipos de filtro.
* Area onde aparece o resultado da busca, contendo a imagem do perfil do aluno, o nome, curso e e-mail.
* Botão “Visualizar”: Redireciona para a página que visualiza o perfil completo do aluno.
* Botão “Editar”: Redireciona para a página que permite editar os dados do cadastro.
* Botão “Excluir”: Botão que permite depois de uma confirmação excluir o usuário cadastrado.
* Apenas o admin pode visualizar essa página.

  
Figura 7 - Protótipo da Tela do Painel do Admin.

* **Tela de cadastro do monitor:**

Permitir que o admin cadastre novos usuários monitores no FatecConnect.Pode ser acessada pelo botão "Cadastrar Monitor" na tela do Painel do Admin.

Funcionalidades e regras:

* Campos obrigatórios: Nome, E-mail, Senha Temporária.
* O campo de e-mail deve aceitar apenas endereços válidos (@fatec.sp.gov.br).
* A senha deve ter no mínimo 6 caracteres e deve coincidir com a confirmação de senha.
* Botão "Cadastrar" para finalizar o registro.
* Apenas o usuário admin pode acessar essa tela para criar uma conta de monitor.

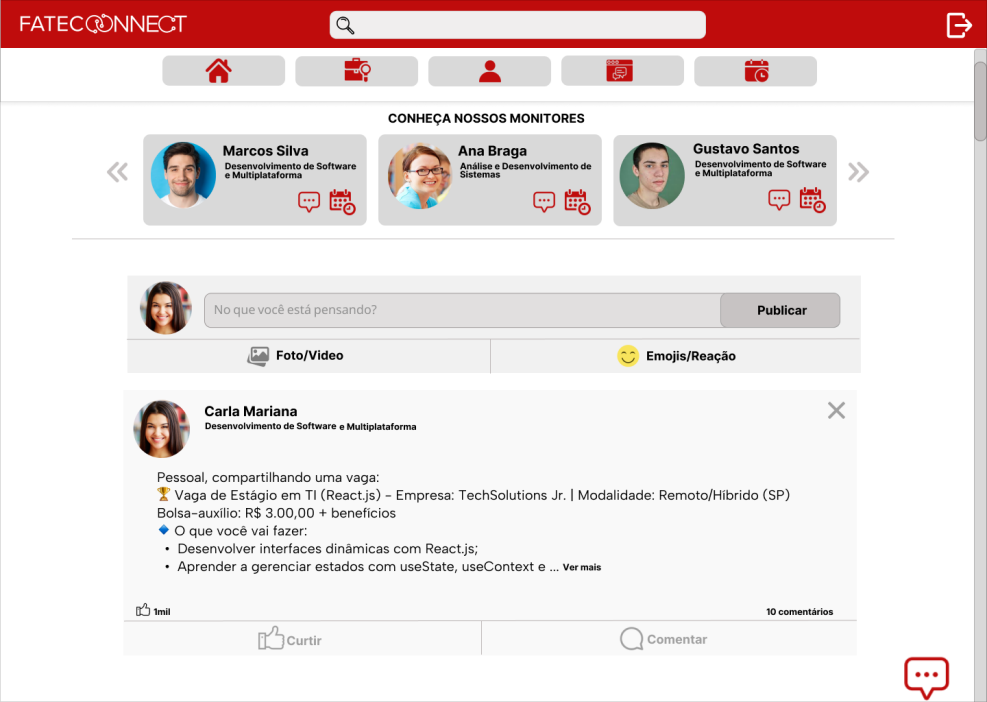
  
Figura 8 - Protótipo da Tela de Cadastro do Monitor

* **Página de postagem do fórum**

Permitir que os usuários postem dúvidas, compartilhem conhecimentos e interajam por meio de comentários. Disponível para usuários autenticados, acessível pelo menu de navegação.

Funcionalidades e regras:

* Campo de busca para encontrar tópicos específicos.
* Lista de postagens exibindo título, autor, curso e imagens/vídeos.
* Botão "Publicar" para iniciar uma nova discussão.
* Apenas usuários autenticados podem interagir no fórum.
* Ícones para acessar o chat e outras funcionalidades.
* Quaisquer usuários logados no sistema podem acessar essa página.

  
Figura 9 - Protótipo da Tela de Postagem do Fórum.

* **Página de agendamento de monitoria**

A página de agendamentos tem o intuito de permitir que os alunos cadastrados

no site possam escolher o dia e a hora em que queiram fazer a monitoria, assim

facilitando a prática e induzindo alunos a participarem.

Funcionalidades e regras:

* Barra de navegação: Possui botões didáticos que levam até a página inicial, vagas de trabalho, perfil do usuário, chat de mensagens e até a página de agendamentos.
* Seleção de monitores: Onde o aluno pode escolher o monitor que mais se adequa às suas necessidades.
* Calendário: Facilitando os alunos no agendamento do dia da monitoria, deixando visível dias livres, dias ocupados e feriados.
* Seleção de horário: Após a escolha do dia, os horários são configurados de acordo com sua disponibilidade, sendo vermelho os horários indisponíveis e azul o horário selecionado pelo aluno.
* Botão “Agendar”: Após a escolha do dia e do horário, o botão “Agendar” serve para marcar a monitoria e salvar no banco de dados a operação feita.
* Botão "Logout": Permite que o usuário saia do perfil.

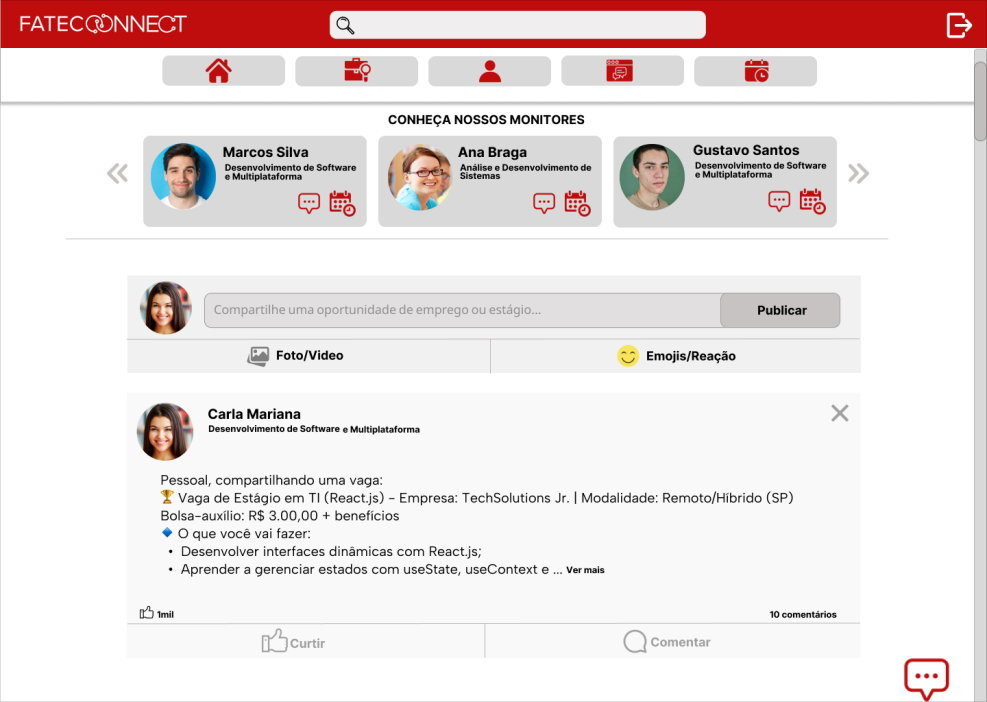
  
Figura 10 - Protótipo da Tela de Agendamento.

* **Página de postagem de vagas**

Essa página tem como o objetivo permitir que os usuários postem vagas encontradas oportunidades e interajam por meio de comentários. Disponível para usuários autenticados, acessível pelo menu de navegação.

Funcionalidades e regras:

* Campo de busca para encontrar tópicos específicos.
* Lista de postagens exibindo título, autor, curso e imagens/videos.
* Botão "Publicar" para postar.
* Apenas usuários autenticados podem interagir no fórum.
* Ícones para acessar o chat e outras funcionalidades.
* Quaisquer usuários logados no sistema podem acessar essa página.

  
Figura 11 - Protótipo da Tela de Postagem de Vagas.

* **Chat em tempo real**

Permitir comunicação em tempo real entre usuários e listar contatos disponíveis. Acessível pelo menu principal após login.

Funcionalidades e regras:

* Exibição de contatos disponíveis para conversa.
* Interface para troca de mensagens em tempo real.
* Suporte a mensagens de texto e emojis.
* Usuários logados.
* Acesso aos usuários que estão online no site.

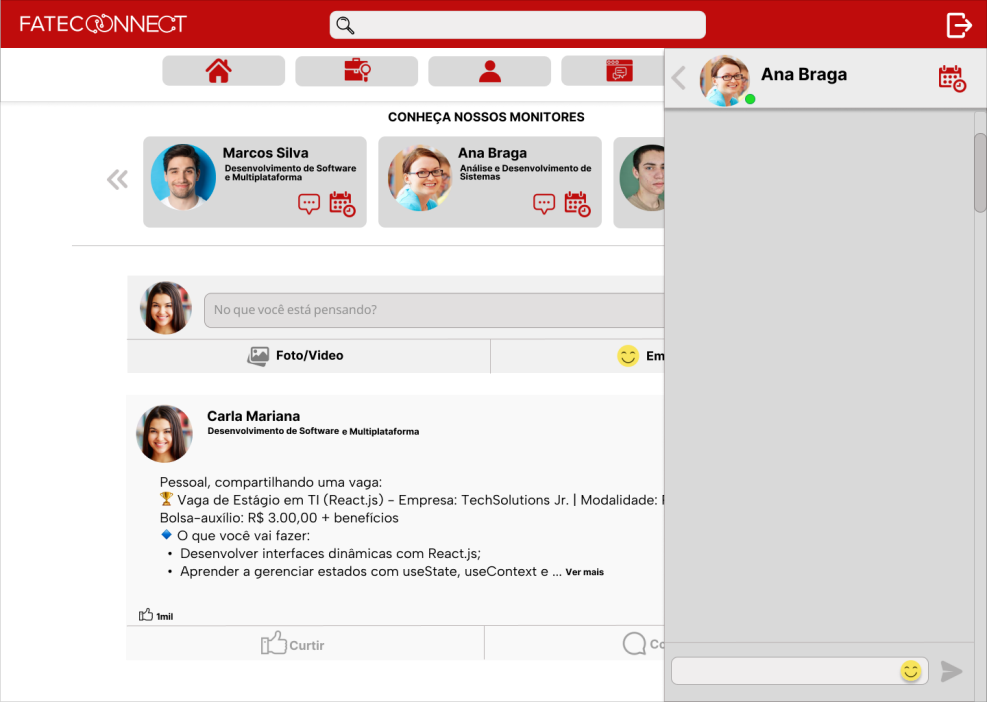
  
Figura 12 - Protótipo da Tela de Postagem do Forúm com demonstração do chat em tempo real.

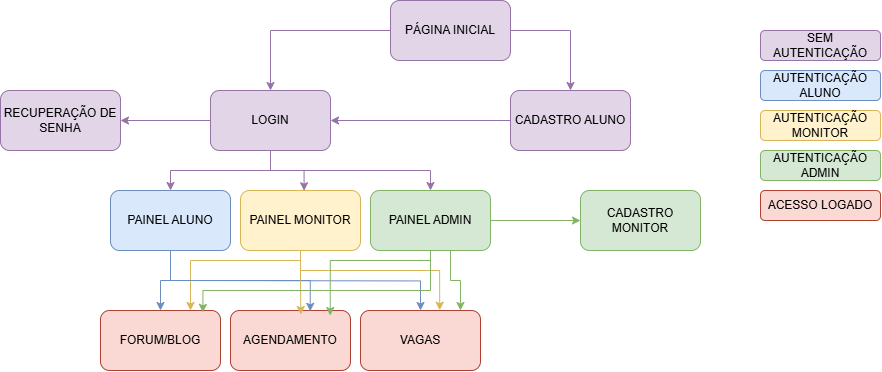


Figura 13 - Protótipo da Tela de Postagem do Forúm com demonstração do chat em tempo real com os contatos.

## 3.3.1 Diagrama de Navegação

O Diagrama de Navegação é uma representação visual da estrutura de navegação do site, demonstra como as páginas estão interligadas e como os usuários podem se mover de uma para outra dentro do site.

* Diagrama das navegações do site FatecConnect:

  
Figura 14 - Navegação de telas.

# 4. ARQUITETURA DO SISTEMA

Este sistema adota uma arquitetura baseada em três camadas principais: Front-End, Back-End e Banco de Dados, seguindo os princípios de separação de responsabilidades e comunicação via API RESTful. Além disso, o sistema oferece chat em tempo real para interação entre os usuários, utilizando tecnologias modernas e escaláveis.

## 4.1. Componentes Principais

#### **• Front-End:** A interface do usuário foi desenvolvida com a bibliotecaReact.js, proporcionando uma experiência interativa e responsiva.

**Tecnologias e bibliotecas utilizadas:**

* **React.js**: Biblioteca JavaScript para construção de interfaces.
* **React Router DOM**: Para navegação entre as páginas da aplicação.
* **Axios**: Para consumo da API de forma assíncrona.
* **Bootstrap**: Framework CSS para estilização rápida e responsiva.

#### • **Back-End:** A API RESTful foi desenvolvida utilizando Node.js e o framework Express, responsável por gerenciar toda a lógica de negócio, autenticação e integração com o banco de dados.

**Tecnologias utilizadas:**

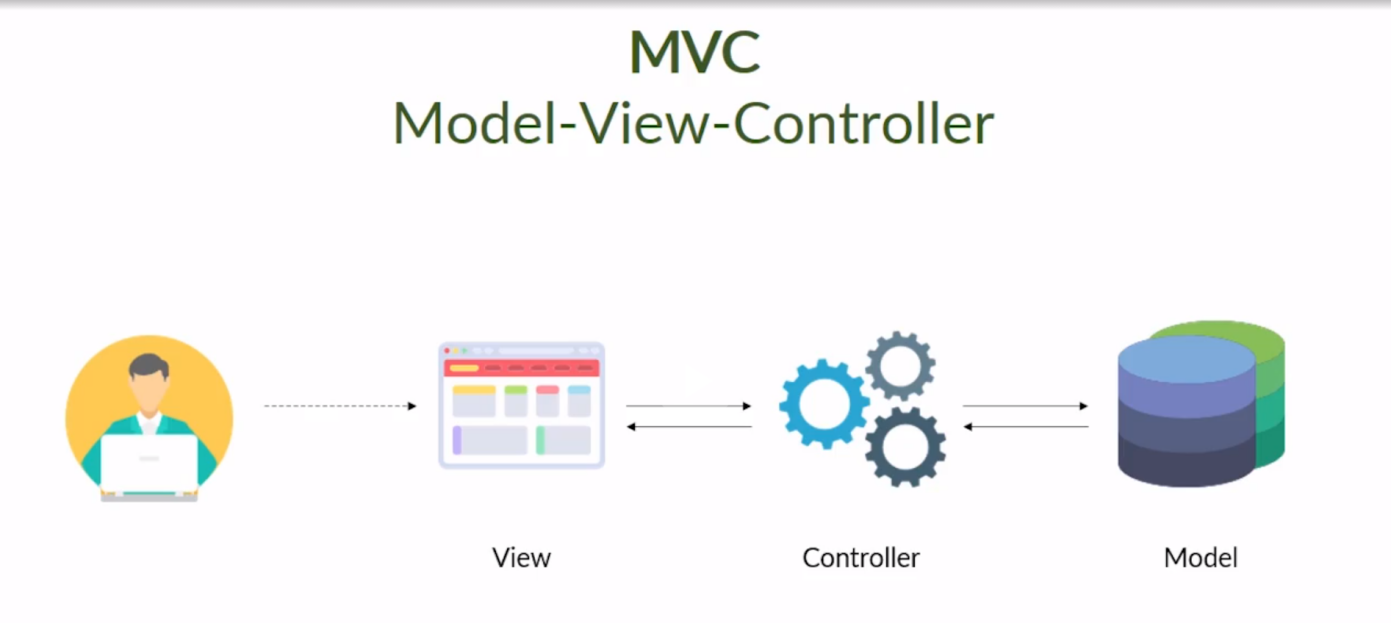
* **Node.js**: Ambiente de execução JavaScript no servidor.
* **Express**: Framework minimalista e flexível para criação de servidores HTTP.
* **MongoDB**: Banco de dados NoSQL para armazenar dados de usuários, tarefas e mensagens.
* **Mongoose**: ODM para modelar objetos de aplicação com o MongoDB.
* **JWT (JSON Web Tokens)**: Utilizado para autenticação e autorização dos usuários.
* **Bcrypt.js**: Para criptografia segura das senhas dos usuários.
* **Nodemailer**: Biblioteca para envio de e-mails (ex: recuperação de senha).
* **CORS**: Middleware para permitir requisições de diferentes origens (como front-end e back-end em servidores distintos).
* **Banco de Dados**: O sistema utiliza o MongoDB como banco de dados, armazenando todas as informações de forma estruturada e orientada a documentos, com coleções para usuários, tarefas e mensagens de chat.

## 4.2. Diagrama de Arquitetura

Este projeto utiliza a arquitetura MVC (Model-View-Controller), um padrão amplamente adotado no desenvolvimento de aplicações web por oferecer uma clara separação de responsabilidades, facilitando a manutenção, escalabilidade e organização do código.

**MVC (Model-View-Controller):**

* **Model (Modelo)**: Responsável pela estrutura dos dados e regras de negócio. No projeto, os modelos são definidos com Mongoose, que representa os dados armazenados no MongoDB.
* **View (Visão)**: Responsável pela interface com o usuário. No momento, o front-end está em fase inicial e utilizará Bootstrap para a criação das interfaces.
* **Controller (Controlador)**: Camada intermediária entre Model e View. No back-end com Express, os controllers recebem as requisições HTTP, processam a lógica e retornam as respostas apropriadas.



## 4.3. Segurança

* Autenticação baseada em JWT.
* Criptografia de senhas com Bcrypt.js.
* Validação de dados de entrada como o e-mail institucional.
* Uso de variáveis de ambiente como dotenv para proteger informações sensíveis.

## 4.4. Escalabilidade

* Banco NoSQL (MongoDB) com suporte a replicação e particionamento.
* Cache pode ser implementado futuramente (ex.: Redis) para otimizar leitura de dados recorrentes.

## 4.5. Manutenção

## Estrutura modular com separação entre rotas e controladores.

* Uso de nodemon no ambiente de desenvolvimento para recarregar automaticamente o servidor após alterações.
* Facilidade de atualização da API sem impacto direto no front-end.

## 4.6. Integração

* **Google Calendar API:** Para os alunos poderem sincronizar as monitorias agendadas com seus calendários pessoais.
* **Zoom / Google Meet API:** Integração para gerar links automáticos de reuniões ao marcar uma aula de monitoria. Ex: O aluno marcou uma monitoria online, o link da reunião já é gerado automaticamente

## 4.7. Limitações

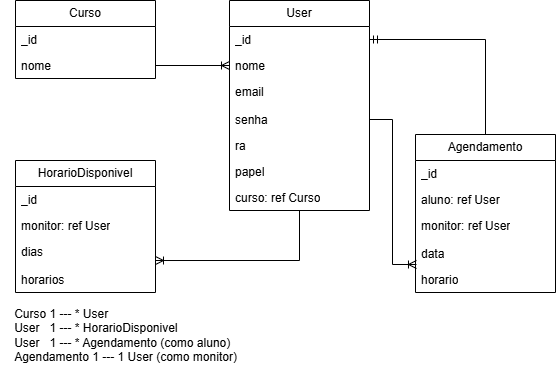
* O uso de MongoDB, apesar de flexível, pode apresentar dificuldades em consultas muito complexas, especialmente se houver crescimento grande de dados sem índices bem planejados.
* O uso de múltiplas bibliotecas sem uma estrutura bem definida (por exemplo, separação clara entre camadas de serviços e controllers) pode tornar a manutenção difícil à medida que o projeto cresce.
* O back-end com Express/Node.js é ótimo para aplicações leves a médias, mas pode ter problemas de desempenho sob alta carga, se não for escalado horizontalmente com balanceamento de carga.
* A falta de foco em acessibilidade (a11y) pode limitar o uso da aplicação por pessoas com deficiências.

# 5. ARQUITETURA DE DADOS

Este capítulo apresenta os fundamentos e diretrizes para a estruturação e o gerenciamento dos dados do sistema FatecConnect, abordando o modelo lógico, a criação física do banco de dados e o dicionário de dados correspondente. Servindo como referência para os desenvolvedores e usuários do sistema.

## 5.1. Modelo Lógico da Base de Dados

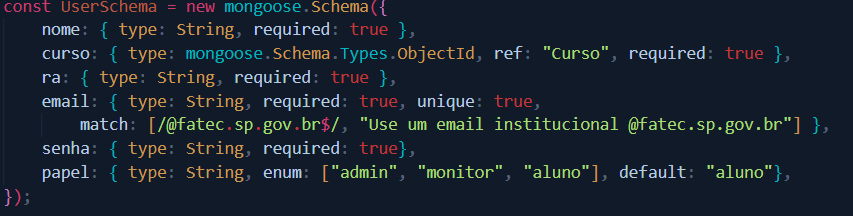
O modelo lógico foi desenvolvido utilizando o paradigma de banco de dados orientado a documentos (NoSQL), por meio da biblioteca Mongoose para MongoDB. A estrutura foi representada por meio de um **Diagrama de Classes**, por ser mais adequado à arquitetura orientada a objetos utilizada.



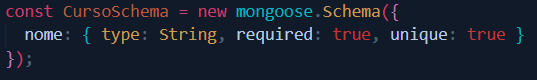
## 5.2. Criação Física do Modelo de Dados

A criação física foi realizada através de esquemas Mongoose, definindo os documentos e seus atributos, bem como as referências entre eles.

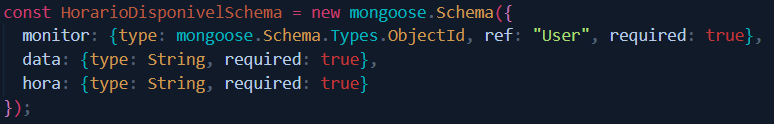
Script: User.js



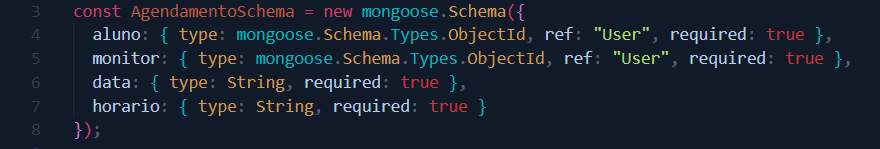
Script: Curso.js



Script: HorarioDisponivel.js



Script: Agendamento.js



## 5.3. Dicionário de Dados

Tabela: User

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome da coluna** | **Tipo de Dado** | **Restrições** | **Descrição** |
| nome | String | Obrigatório | Nome do usuario |
| curso | ObjectId(ref: Curso) | Obrigatório | Curso do usuario matriculado |
| ra | String | Obrigatório | Registro academico |
| email | String | Obrigatório, único | E-mail Institucional |
| senha | String | Obrigatório | Senha de login |
| papel | String | Enun: (admin, monitor, aluno). Padrão: aluno | O tipo de perfil do aluno, se é monitor, administrador ou aluno somente |

Tabela: Curso

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome da coluna** | **Tipo de valor** | **Restrição** | **Descrição** |
| nome | String | Obrigatório | Nome do curso |

Tabela: HorarioDisponivel

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome da coluna** | **Tipo de valor** | **Restrição** | **Descrição** |
| monitor | ObjectId(ref: User) | Obrigatório | Nome do monitor com o horário disponível |
| data | String | Obrigatório | Data do horário disponível para agendamento de monitoria |
| hora | String | Obrigatório | Hora do horário disponível para agendamento de monitoria |

Tabela: Agendamento

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome da coluna** | **Tipo de valor** | **Restrição** | **Descrição** |
| aluno | ObjectId(ref: User) | Obrigatório | Nome do aluno que solicitou o agendamento |
| monitor | ObjectId(ref: User) | Obrigatório | Nome do monitor solicitado no agendamento da monitoria |
| data | String | Obrigatório | Data da monitoria |
| horario | String | Obrigatório | Horário marcado da monitoria |

# 6. PROJETO DE SOFTWARE

Este capítulo tem como objetivo detalhar o design interno dos componentes do sistema com base na arquitetura já definida, preparando a equipe para a fase de implementação. Ele detalha como as partes do sistema interagem, quais responsabilidades cada componente tem e como a lógica será distribuída.

## 6.1. Design dos Componentes

O sistema adota uma arquitetura modular, separando responsabilidades para garantir organização, manutenibilidade e escalabilidade.

**Backend (Node.js + Express):**

* **config/db.js:**

Módulo responsável pela conexão com o banco de dados MongoDB, utilizando a biblioteca Mongoose. Define as configurações de URI e exibe mensagens de conexão bem-sucedida ou falha.

* **controller/:**

A pasta controller armazena os controladores, que são os arquivos responsáveis por centralizar a lógica das funcionalidades do sistema.

O controller faz o meio de campo entre as requisições que chegam pelas rotas (routes) e as interações com o banco de dados (models).

* **models/:**

A pasta models armazena os modelos de dados(schemas) da aplicação, definidos com o Mongoose, que é a biblioteca usada para modelar objetos do MongoDB como User, Curso, Agendamento.

* **middleware/:**

Contém middlewares que são funções executadas antes dos controladores das rotas **verificarToken.js** e **verificarAdmin.js.**

* **verificarToken.js:** Verifica se o JWT fornecido no cabeçalho Authorization é válido. Se sim, anexa os dados do usuário à requisição.
* **verificarAdmin.js:** Verifica se o usuário autenticado tem o papel admin, restringindo certas rotas apenas para administradores.

* **routes/:**

A pasta routes é responsável por definir as rotas da aplicação, ou seja, os caminhos HTTP (como /login, /agendamentos, /usuarios, etc.) e os métodos associados a eles (GET, POST, PUT, DELETE).

* **node\_modules/:**

Contém todas as dependências do projeto, como express, mongoose, bcryptjs, jsonwebtoken, dotenv, entre outras. Não é modificada manualmente.

* **.env:**

Armazena variáveis sensíveis, como a URI do banco de dados e a chave JWT.

* **package.json e package-lock.json:**

Gerenciam as dependências, scripts e metadados do projeto.

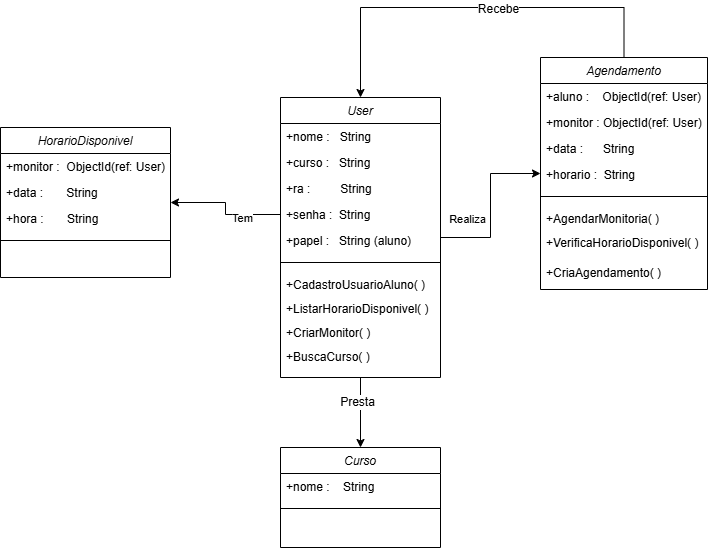
* **server.js:**

Arquivo principal que inicializa o servidor Express, configura middlewares, conecta ao banco (via config/db.js) e define as rotas (authRoutes.js).

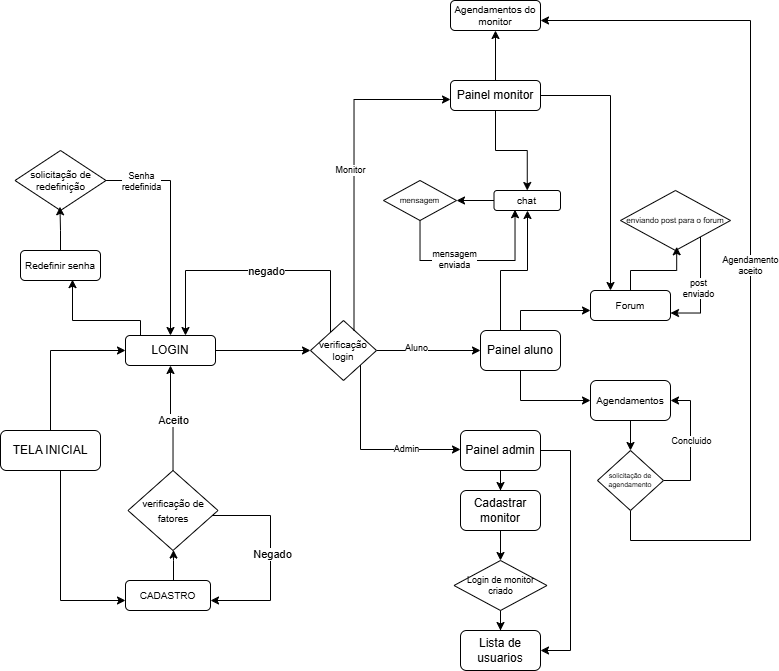
**Frontend (React.js):**

* **Páginas:** Representam cada tela (Tela inicial, Login, Cadastro, Painel Aluno/Monitor/Admin, Fórum, Agendamento, Chat).
* **Componentes:** Reutilizáveis (Fórum, Navbar, Painel).
* **Serviços (API):** Comunicação com o backend via Axios.
* **Gerenciamento de Estado:** Utiliza useContext e useState.

## 6.2. Diagrama de Classes



## 6.3. Diagrama de Sequência ou Fluxo de Interações



* **Fluxo de Login com verificação e redirecionamento por tipo de usuário.**

**1.** Usuário acessa a tela inicial.

**2.** Caso já tenha uma conta, o usuário clica em “Login” e insere suas credenciais.

**3.** O sistema realiza a **verificação de fatores** (validação de dados, e-mail/senha).

**4.** Se **verificação for negada**, o acesso é recusado.

**5.** Se for **aceita**, ocorre a **verificação de login**:

- Se for **Aluno**, redireciona para o **Painel Aluno**.

- Se for **Monitor**, redireciona para o **Painel Monitor**.

- Se for **Admin**, redireciona para o **Painel Admin**.

* **Fluxo de Cadastro de novo Usuário.**

**1.** Na tela inicial o usuário opta por se cadastrar

**2.** Insere os dados exigidos (nome, e-mail institucional, curso e senha).

**3.** O sistema realiza a **verificação de fatores** (validação de dados, e-mail/senha).

**4.** Se a verificação for bem-sucedida, o cadastro é aceito e o usuário pode seguir para o login.

**5.** Caso contrário, o acesso é negado e o usuário é orientado a corrigir os dados.

* **Fluxo de agendamento de monitoria (Aluno para Monitor).**

**1.** Aluno acessa o **Painel Aluno**.

**2.** Vai até a seção de **Agendamentos** e realiza uma **solicitação de agendamento** com um monitor.

**3.** A solicitação é processada e exibida no **Painel do Monitor** correspondente.

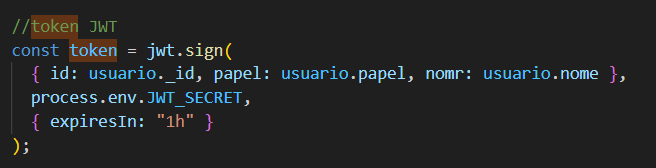
**4.** O monitor pode aceitar o agendamento.

**5.** Quando aceito, o agendamento é marcado como **concluído** no sistema.

## 6.4. Regras de Negócio Detalhadas

A aplicação FatecConnect implementa diversas regras de negócio fundamentais para garantir a **segurança**, **integridade dos dados** e o **controle de acesso** de acordo com o papel do usuário no sistema. A seguir, estão detalhadas essas regras:

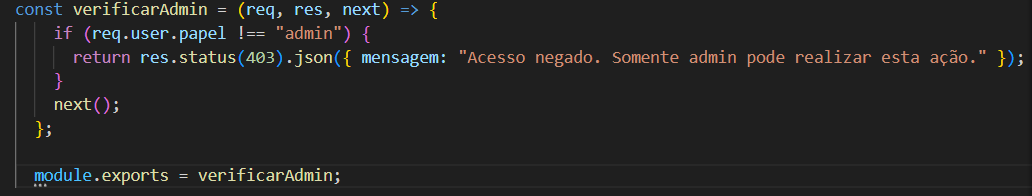
* **Autenticação com JWT:** Todo usuário que realiza login com **email e senha válidos** recebe um token JWT contendo seu \_id e papel. Esse token é necessário para acessar rotas protegidas.

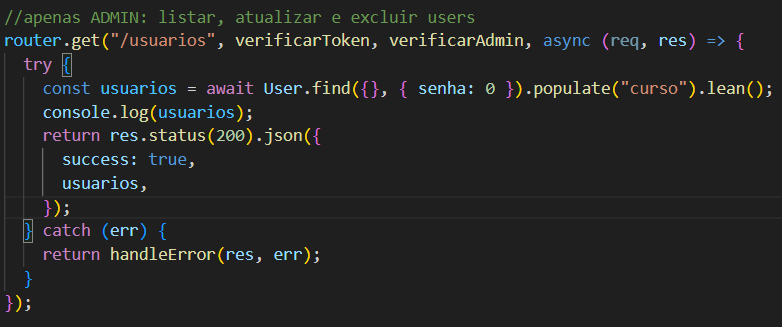


- O token expira em 1 hora, exigindo novo login após esse período (expiresIn: '1h'). Todos os tokens JWT expiram após 1 hora, reforçando a segurança da aplicação.

- É utilizado para autenticar e autorizar usuários em endpoints protegidos via middleware *verificarToken*.

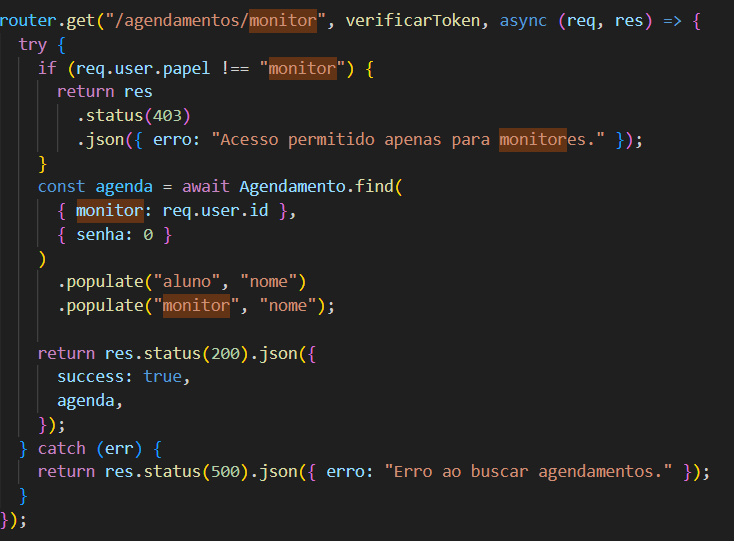
* **Controle de Acesso por Papel (Role-Based Access Control):**
* **Admin:** Admins têm acesso irrestrito a todas as rotas e funcionalidades críticas do sistema, como gerenciamento de usuários e visualização de dados completos.





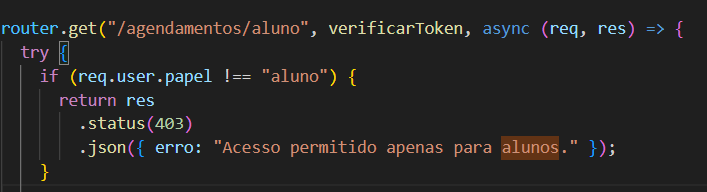
Somente admin pode cadastrar um monitor e/ou editar informações ou excluir usuários.

* **Monitores:** Monitores podem criar e gerenciar sua agenda de monitorias, além de visualizar os agendamentos feitos por alunos para suas sessões.



Essa parte do código é responsável por listar os agendamentos vinculados ao monitor logado. A rota é protegida por um middleware de autenticação JWT e valida se o usuário tem papel de "monitor". Se a verificação for positiva, ela busca todos os agendamentos cujo campo monitor corresponde ao id do usuário logado. A resposta inclui os dados populados do aluno e do monitor para facilitar a exibição no front-end.

- **Alunos:** Alunos têm permissões limitadas apenas às suas ações como agendar sessões de monitoria com monitores disponíveis e só podem visualizar seus próprios dados e agendamentos.



* **Agendamento de Monitoria:** Alunos podem solicitar agendamentos com monitores disponíveis, o sistema verifica se o monitor e o aluno existem, se a data enviada é válida e se não há conflito de horário. Todos os agendamentos são criados com status inicial “pendente” e podem ser atualizados para “concluído” ou “cancelado”.
* **Status do agendamento:** apenas monitores e admins podem atualizar o status de um agendamento. Mas alunos podem cancelar seus próprios agendamentos.

## 6.5. Estratégias de Tratamento de Erros

**Backend (Node.js):**

* **Try/Catch:** Utilizado em todos os controllers para capturar exceções.
* **Middleware de Erros:** Middleware global trata erros inesperados e retorna mensagens com status apropriado (400, 401, 404, 500).
* **Validação JWT:** Se o token estiver ausente ou inválido, retorna 401 Unauthorized.
* **Permissões:** Se o usuário tentar acessar uma funcionalidade sem o papel necessário, retorna 403 Forbidden.

**Frontend (React):**

* **Axios:** Captura erros com try/catch nas chamadas de API.
* **Mensagens e alertas:** Exibição de alertas ou mensagens como "Usuário ou senha inválidos", "Erro ao agendar monitoria".
* **Redirecionamento:** Em caso de erro 404, o usuário é redirecionado para uma página de erro personalizada.

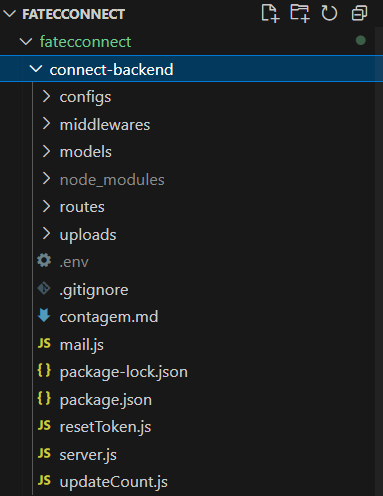
# 7. IMPLEMENTAÇÃO

Este capítulo documenta como as funcionalidades do sistema foram implementadas utilizando **Node.js no back-end**, **React no front-end** e **MongoDB Atlas como banco de dados**, detalhando a estrutura da aplicação, as interações entre os componentes e a aplicação de conceitos de **modularidade**, **reutilização de código** e **boas práticas de desenvolvimento web**.

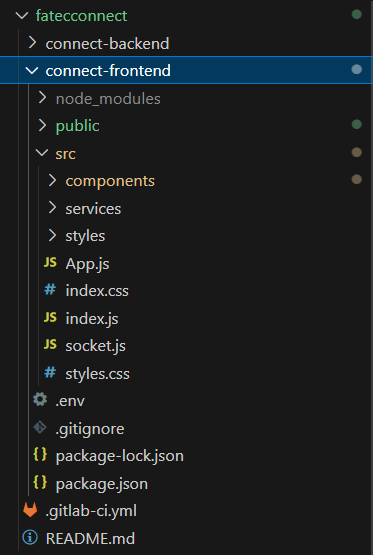
## 7.1. Estrutura geral do projeto

A estrutura do projeto foi organizada de forma modular, separando claramente o front-end (interface do usuário) do back-end (API e lógica de negócio). No back-end, rotas, controladores e modelos foram distribuídos de forma clara, permitindo manutenibilidade e escalabilidade.

* Estrutura pasta Backend:



* Estrutura pasta Front-end:

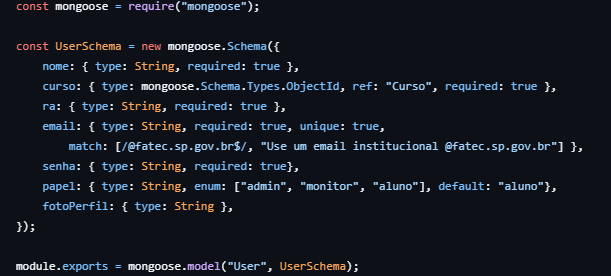


## 7.2. Modelos

Os modelos são arquivos que definem a estrutura dos documentos nas coleções do banco de dados MongoDB. Utilizando o Mongoose (ODM do Node.js), cada modelo representa uma coleção e define os campos, tipos de dados, validações e relacionamentos. Esses modelos permitem criar, ler, atualizar e deletar documentos de forma prática e organizada, servindo como base para a manipulação dos dados em toda a aplicação.

### 7.2.1 Modelo User

A classe User representa os usuários do sistema, que podem ser administradores, monitores ou alunos. Ela armazena dados essenciais para autenticação, identificação acadêmica e controle de acesso com base em papéis.



* **Campos:**  nome, curso, ra, email, senha, papel.
* **Métodos: CadastroUsuarioAluno( )** : Cadastra um novo aluno

**ListarHorarioDisponivel( )** : Lista os horarios disponivéis para monitoria

**CriarMonitor( )** : Cria um User monitor, apenas um user admin pode ter acesso a esse método

**BuscaCurso( )** : Busca o curso pelo nome. Apenas um User admin tem acesso a esse método.

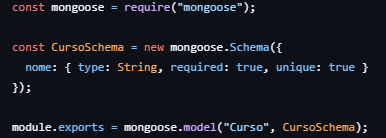
**AtualizaUser( )** : Atualiza informações do User. Apenas um User admin tem acesso a esse método

**DeletaUser( )** : Deleta um User. Apenas um User admin tem acesso a esse método.

**ListaUser( )** : Lista os User existentes. Apenas um User admin tem acesso a esse método

### 7.2.2 Modelo Curso

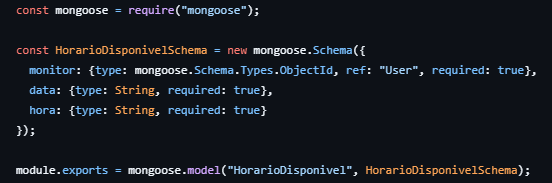
A classe Curso representa os cursos disponíveis na instituição Fatec, sendo associada aos usuários que pertencem a esse curso.



* **Campos:**  nome
* **Métodos: Nenhum método para essa classe**

### 7.2.3 Modelo HorarioDisponivel

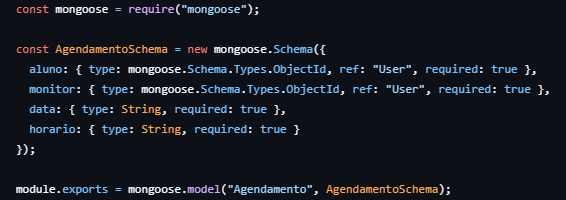
A classe HorarioDisponivel representa os horários em que um monitor está disponível para atendimento da monitoria.



* **Campos:**  monitor, data, hora.
* **Métodos: Nenhum método para essa classe**

### 7.2.4 Modelo Agendamento

A classe Agendamento representa a reserva de data feita em relação a monitoria, de aluno com um monitor.



* **Campos:** aluno, monitor, data, horario.
* **Métodos: AgendarMonitoria( )** : Agenda a monitoria entre monitor e aluno

**VerificaHorarioDisponivel( )** : Verifica os horários disponíveis para agendamento de uma monitoria

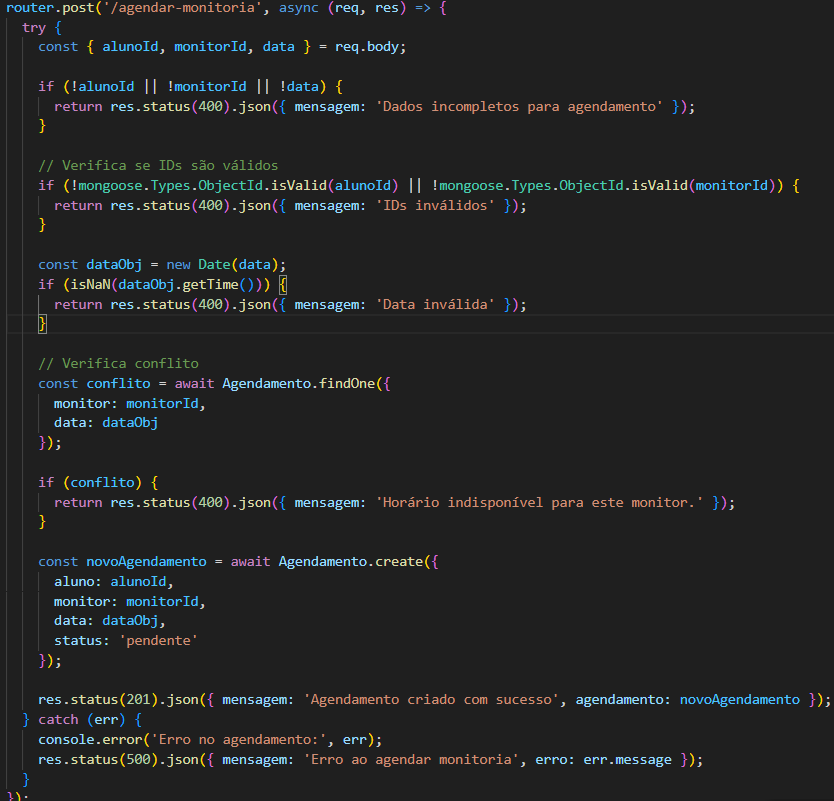
**CriaAgendamento( )** : Cria o agendamento

## 7.3. Views (Routes)

As views no projeto *FatecConnect são as* ***Routes***, construídas com **Node.js e Express**, são responsáveis por tratar as requisições HTTP da aplicação. Elas acessam os modelos MongoDB (via Mongoose), realizam validações e regras de negócio, e retornam as respostas em formato JSON para o front-end em React.

### 7.3.1. View para Criar Agendamento

A view (Routes) POST /agendar-monitoria permite que um aluno agende uma monitoria com um monitor, verificando previamente se o horário está disponível.



* **Validação de entrada:** verifica se os campos alunoId, monitorId e data foram enviados e se são válidos.
* **Conversão da data:** converte a data enviada para um objeto Date, verificando se é válida.
* **Verificação de conflito:** consulta o banco para garantir que o monitor não tem outro agendamento na mesma data e horário.
* **Criação do documento:** se tudo estiver válido, cria um novo Agendamento com status inicial "pendente".
* **Resposta:** retorna status 201 Created com os dados do novo agendamento.

## 7.4 Templates (Componentes)

No projeto **FatecConnect**, a interface visual da aplicação é construída utilizando **Componentes React**. Esses componentes são responsáveis por exibir dados dinâmicos na tela, como perfis de usuário, mensagens, agendamentos e postagens, além de permitir a interação do usuário com o sistema.

Cada componente React é composto por uma estrutura HTML, com algumas estilizações específicas feitas em CSS — sendo a maior parte implementada com a biblioteca **Bootstrap** — e lógica em JavaScript para manipulação de dados e eventos. Por meio do uso de **hooks**, como useState e useEffect, os componentes reagem dinamicamente às ações dos usuários e às respostas da API, mantendo a interface atualizada em tempo real.

### 7.4.1 Template para Exibir os Agendamentos



Cada agendamento é renderizado como uma linha da tabela (<tr>), contendo colunas com as seguintes informações:

* **Nome do monitor** (ou "Desconhecido", caso não disponível),
* **Data e hora** do agendamento, formatadas para o padrão brasileiro,
* **Status** do agendamento (ex: "pendente", "cancelado", "concluído"),
* **Ação de cancelamento**, exibida como um ícone clicável (caso o status não seja "cancelado" ou "concluído").

**REFERÊNCIAS**

META. React – A JavaScript library for building user interfaces. 2025. Disponível em: [https://react.dev/](https://react.dev/" \t "_new). Acesso em: 9 jun. 2025.

REMIX SOFTWARE. React Router DOM – Declarative routing for React apps. 2025. Disponível em: [https://reactrouter.com/](https://reactrouter.com/" \t "_new). Acesso em: 9 jun. 2025.

BOOTSTRAP. Bootstrap – The most popular HTML, CSS, and JS library in the world. 2025. Disponível em: [https://getbootstrap.com/](https://getbootstrap.com/" \t "_new). Acesso em: 9 jun. 2025.

MUI. Material UI – React UI Framework. 2025. Disponível em: [https://mui.com/](https://mui.com/" \t "_new). Acesso em: 9 jun. 2025.

AXIOS. Axios – Promise based HTTP client for the browser and node.js. 2025. Disponível em: [https://axios-http.com/](https://axios-http.com/" \t "_new). Acesso em: 9 jun. 2025.

SOCKET.IO. Socket.IO – Realtime application framework. 2025. Disponível em: [https://socket.io/](https://socket.io/" \t "_new). Acesso em: 9 jun. 2025.

OPENJS FOUNDATION. Node.js. 2025. Disponível em: [https://nodejs.org/](https://nodejs.org/" \t "_new). Acesso em: 9 jun. 2025.

EXPRESS. Express – Fast, unopinionated, minimalist web framework for Node.js. 2025. Disponível em: [https://expressjs.com/](https://expressjs.com/" \t "_new). Acesso em: 9 jun. 2025.

MONGODB INC. MongoDB – The developer data platform. 2025. Disponível em: [https://www.mongodb.com/](https://www.mongodb.com/" \t "_new). Acesso em: 9 jun. 2025.

MONGOOSE. Mongoose – Elegant MongoDB object modeling for Node.js. 2025. Disponível em: [https://mongoosejs.com/](https://mongoosejs.com/" \t "_new). Acesso em: 9 jun. 2025.

AUTH0. JWT – JSON Web Tokens. 2025. Disponível em: [https://jwt.io/](https://jwt.io/" \t "_new). Acesso em: 9 jun. 2025.

BCYRPT.JS. Bcryptjs – Optimized bcrypt in JavaScript with zero dependencies. 2025. Disponível em: [https://github.com/dcodeIO/bcrypt.js](https://github.com/dcodeIO/bcrypt.js" \t "_new). Acesso em: 9 jun. 2025.

NODEMAILER. Nodemailer – Send e-mails with Node.js. 2025. Disponível em: [https://nodemailer.com/](https://nodemailer.com/" \t "_new). Acesso em: 9 jun. 2025.

CASTRO, Bruno. Como fazer um CHAT EM TEMPO REAL utilizando WEBSOCKETS com REACT E NODE.JS. YouTube, 2023. Disponível em: [https://www.youtube.com/watch?v=Yi2crLU9WA0](https://www.youtube.com/watch?v=Yi2crLU9WA0" \t "_new). Acesso em: 9 ABR. 2025