

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS  
ESCOLA POLITÉCNICA  
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**



**Aplicação Web para Avaliação de Bares e Restaurantes**

**CRISTIANO IRIAS VIEIRA FILHO**

**GOIÂNIA  
2022**

**CRISTIANO IRIAS VIEIRA FILHO**

**Aplicação Web para Avaliação de Bares e Restaurantes**

**GOIÂNIA  
2022**

**CRISTAINO IRIAS VIEIRA FILHO**

**Aplicação Web para Avaliação de Bares e Restaurantes**

**GOIÂNIA  
2022**

## RESUMO

Como parte do avanço do desenvolvimento, surgiram ferramentas que facilitam em grande escala a produção de aplicações web feitas por uma equipe ou até mesmo para um único desenvolvedor. Dentro dessas ferramentas, existem os *frameworks*, e como principais temos os derivados do *JavaScript*, que são *React*, *Angular* e *Vue*. Outra grande ferramenta usada para facilitar a vida de desenvolvedores é o *Node.js* que tem a funcionalidade de ser *server-side* de uma aplicação, sem necessitar de um browser para execução. Dito isso, foram usados *React* e *Node.js* para o desenvolvimento de uma ferramenta web para avaliação de bares e restaurantes. O intuito desse trabalho é avaliar as facilidades e funcionalidades dos frameworks utilizados que permitem a produção de uma aplicação web de alta qualidade.

**Palavras-chave:** *React*, *Node.js*, desenvolvimento web, *javascript*

## **ABSTRACT**

As part of the advancement of development, tools have emerged that greatly facilitate the production of web applications made by a team or even a single developer. Within these tools, there are frameworks, and as main ones we have the JavaScript-derived ones, which are React, Angular, and Vue. Another great tool used to make developers' lives easier is NodeJs which has the functionality of being server-side of an application, without needing a browser for execution. That being said, React and Node.js were used to develop a web tool for bar and restaurant reviews. The intention of this work is to evaluate the facilities and functionalities of the frameworks used that allow the production of a high quality web application.

**Keywords:** *React, Node.js, web development, javascript*

## LISTA DE ILUSTRAÇÕES

Figura 1. Representação da Arquitetura da Aplicação.....	17
Figura 2. Tela de Login.....	19
Figura 3. Tela Principal.....	19
Figura 4. Tela de Alocação de Salas .....	20
Figura 5. Tela VS Code Loja de Extensões.....	23
Figura 6. Caso de Usuário Bar .....	24
Figura 7. Caso de Uso Usuário Cliente .....	25
Figura 8. Tela de Cadastro Usuário cliente.....	26
Figura 9. Tela de Login do Usuário Cliente.....	27
Figura 10. Tela Principal da Aplicação.....	27
Figura 11. Tela de Realizar Avaliação .....	28
Figura 12. Tela de Edição de Perfil .....	29
Figura 13. Tela de Cadastro do Usuário Bar .....	30
Figura 14. Tela de Avaliações Recebidas pelo Usuário Bar .....	31
Figura 15. Modelo Conceitual do Banco.....	32

## LISTA DE SIGLAS

API	<i>Application Program Interface</i>
CSS	<i>Cascading Style Sheets</i>
DOM	<i>Document Object Model</i>
ES5	<i>ECMAScript 2009</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hyper Text Protocol</i>
HTTPS	<i>Hyper Text Protocol Secure</i>
IDE	<i>Integrated Development Environment</i>
I/O	<i>Input And Output</i>
JS	<i>JavaScript</i>
JSX	<i>JavaScript Sintax Extension</i>
NoSql	<i>Not Only Sql</i>
NPM	<i>Node Packager Manager</i>
PHP	<i>Hypertext Preprocessor</i>
SQL	<i>Standard Query Language</i>
VDOM	<i>Virtual Document Object Model</i>
VSCoDe	<i>Visual Studio Code</i>
XML	<i>eXtensible Markup Language</i>
WEB	<i>World Wide Web</i>

# Sumário

<b>1 INTRODUÇÃO</b>	9
1.1 Objetivo geral e específicos	11
1.2 Justificativa	11
1.3 Resultado Esperado	12
1.4 Estrutura do Trabalho	12
<b>2 REVISÃO BIBLIOGRÁFICA</b>	13
2.1 ReactJS	13
2.2 JSX	13
2.3 Node.js	14
2.4 Node.js Express	15
2.5 MongoDB	15
2.6 Arquitetura da Aplicação	16
<b>3 TRABALHOS RELACIONADOS</b>	18
3.1 Sistema de Alocação de Salas	18
3.2 Estudo sobre desenvolvimento de aplicações web com a linguagem javascript e a utilização do framework React	20
<b>4 AMBIENTE DE PRODUÇÃO E METODOLOGIA</b>	22
4.1 Ambiente de desenvolvimento	22
4.2 Descrição da aplicação	24
4.3 Diagrama do banco de dados	31
<b>5 RESULTADOS</b>	33
<b>6 REFERÊNCIAS</b>	35



# 1 INTRODUÇÃO

De acordo com estudo realizado em uma parceria da *Hoopsuite* com a *We Are Social* em 2019, em média, o brasileiro passa nove horas e vinte nove minutos conectados à internet todos os dias. A pesquisa coloca o Brasil apenas atrás das Filipinas, que passa dez horas por dia conectada à internet. (CANALTECH, 2019)

Com isso, tem-se que boa parte do ano, o brasileiro passa em contato direto com aplicações *web* ou *mobile*, seja ela uma rede social, um serviço de *stream*, um site de vendas ou qualquer outro tipo de ferramenta. Ao todo, são aproximadamente, 197 dias. Para que esse contato exista, há uma necessidade de se desenvolver tais plataformas, de forma que elas estejam disponíveis ao usuário, de maneira fácil e de acesso imediato. Tais aplicações, na Área da Computação, são rapidamente desenvolvidas através da Programação WEB. (TECHTUDO, 2022)

O desenvolvimento *WEB* foca na construção de sites, *software*, bancos de dados e vários outros serviços que podem ser encontrados na Internet. A desenvoltura dessa atividade específica, sugere como prática de programação, o uso de ferramentas, frameworks de desenvolvimento que utilizam metodologias ágeis de produção. Diante deste contexto, tem-se ferramentas de produção da interface gráfica da aplicação com o usuário, formalmente denominadas *front-end*, bem como um conjunto elaborado de aplicações especificamente desenvolvidas para a manipulação e controle de dados das aplicações. Tal conjunto de aplicações é formalmente conhecido por *back-end*. (VOOMP, 2021)

Na arquitetura cliente-servidor, há dois lados:

- o *client-side*: local que hospeda o *front-end*.
- o *server-side*: local que mantém o *back-end*.

Em se tratando de atividades formais de desenvolvimento para WEB, o desenvolvedor *front-end* é responsável pela experiência do usuário dentro de uma aplicação *web*. É ele quem vai desenhar e desenvolver páginas com as quais, posteriormente, o usuário irá interagir. O *front-end* inclui elementos que determinam a identidade visual de um site ou aplicação. (DIGITAL HOUSE, 2019)

Em contrapartida, o *back-end*, ou *server-side*, faz referência ao que é processado no servidor, por meio de diferentes linguagens. Um desenvolvedor *back-end* além de saber dominar linguagens específicas, também precisa reter conhecimento sobre servidores de redes de computadores, pois são lá que as operações *back-end* acontecem. É também de fundamental importância, que esse profissional também conheça bem a teoria de banco de dados, pois as informações da aplicação ficam armazenadas aqui. Finalmente, o profissional deve desenvolver habilidades referentes à produção de um mecanismo importante para o funcionamento da aplicação, a denominada API (*Application Programming Interface*), que tem o papel de conectar dois sistemas por meio de uma linguagem de programação comumente usada no *back-end*. (DIGITAL HOUSE, 2019)

O site Xpeducacao fez uma pesquisa mostrando que as principais linguagens hoje utilizadas para o *front-end* em desenvolvimento web é o *JavaScript* em primeiro lugar, seguido por HTML e CSS. Já para o *back-end* as linguagens preferidas são *Python*, *Java* e *PHP*. (XPEDUCAÇÃO, 2022)

Dentro das ferramentas possíveis para *front-end* há *frameworks* que são bibliotecas que contém conjuntos de códigos de alguma linguagem de programação em meio de facilitar a vida de quem desenvolve. Nesse trabalho é usado o *framework React*.

O *React* é o *framework* mais popular dentre os desenvolvedores da atualidade. O *React* utiliza bibliotecas do *JavaScript* e é usada para construir uma interface de usuário (UI). Ela oferece uma resposta rápida para o usuário adicionar comandos usando um novo método de renderizar sites. Os componentes dessa ferramenta foram desenvolvidos pelo *Facebook*. Ela foi lançada em 2013 como uma ferramenta *JavaScript* de código aberto. (HOSTINGER, 2022)

Outra ferramenta selecionada para esse trabalho, é o *Node.js*. *Node.js*, é desenvolvido sobre o motor (*engine*) *JavaScript V8* criada pelo Google e utilizada nos navegadores *Chrome* e *Chromium* (*open source*). A ferramenta leva o processamento e renderização do código *JavaScript* para o lado do servidor

(*back-end*). Isso, aliado ao fato de suas requisições I/O (*input* e *output*) serem não bloqueantes, possibilita criar aplicações rápidas, escaláveis e estáveis.

Dentro desse contexto, a aplicação desenvolvida nesse trabalho, é uma aplicação web, voltada para o ramo de restaurantes e bares, onde o usuário pode avaliar o tal da maneira que lhe for mais adequada. O desenvolvimento foi realizado através do *React* e do *Node.js*.

## 1.1 Objetivo geral e específicos

Esse trabalho tem como objetivo principal desenvolver um aplicativo web direcionado ao ramo de bares/restaurantes para avaliação online do estabelecimento pelos clientes, através do uso do *React* e do *Node.js* como frameworks de desenvolvimento.

Além disso, os seguintes objetivos específicos e necessários foram adotados:

- Aprimorar técnicas sobre desenvolvimento web;
- Aplicar conceitos de engenharia de software para desenvolvimento web;
- Aplicar frameworks atuais de desenvolvimento web (*React* e *Node.js*).

## 1.2 Justificativa

Perante o apresentado até o momento, o presente trabalho justifica-se pelo próprio objetivo: desenvolver uma aplicação web, a partir dos estudos sobre a linguagem *JavaScript* e sobre o framework *React* para a parte do *front-end* e usando *Node.js* na parte de *back-end*. Adotando-se os métodos estudados, foi encontrado a melhor maneira possível, rápida e eficiente para o desenvolvimento desse trabalho, "com a geração satisfatória de resultados, que serão apresentados no Capítulo 05.

### **1.3 Resultado Esperado**

Como resultado esperado, pretende-se desenvolver uma aplicação *WEB* funcional, com o uso do *React* para o *front-end*, o *back-end* em *Node.js*, com consumo de APIs e conexão com o banco de dados *MongoDB*, com o intuito de avaliar aspectos de desenvolvimento rápido de aplicações através das ferramentas selecionadas.

### **1.4 Estrutura do Trabalho**

O trabalho está organizado com a seguinte estrutura: O capítulo 2 descreve a revisão bibliográfica, apresentando as ideias e conceitos sobre *Desenvolvimento web, front-end, back-end e Frameworks*. No capítulo 3, o referencial teórico é apresentado, isto é, os trabalhos relacionados. O capítulo 4 descreve o ambiente de produção e metodologia utilizados. O capítulo 5 apresenta os resultados obtidos, junto com as discussões deles. O capítulo 6, contém a conclusão do trabalho e sugestões para trabalhos futuros.

## 2 REVISÃO BIBLIOGRÁFICA

Neste Capítulo serão apresentadas as tecnologias que serão utilizadas e os conceitos em que será baseado o desenvolvimento deste trabalho. Dentre tais tecnologias já supracitadas, estão o *ReactJS*, *Node.js* e *MongoDB*.

### 2.1 ReactJS

O *ReactJs* é um *framework* descrito como uma popular biblioteca *javascript* usada para construir interface de usuário, podendo ser utilizada tanto no desenvolvimento web quanto no para aplicações mobile. Ela oferece uma resposta excelente para o usuário adicionar comandos usando um novo método de renderizar sites. Isso porque o *React* é voltado ao desenvolvimento de componentes, onde o desenvolvedor pode reutilizar os componentes criados em qualquer aplicação que esteja utilizando *React*. (HOSTINGER, 2022)

A “componentização” do *React* é feito através do JSX, uma extensão de sintaxe opcional para *javascript*, que permite combinar HTML com *javascript*. JSX faz com que a renderização de múltiplas funções seja mais fácil.

Uma grande vantagem de se usar o *React* e um dos motivos da escolha dele nesse trabalho, é que com o *React* é possível a construção de um *Virtual DOM* e que ele seja hospedado na memória. (KENZIE, 2022)

O *Virtual Document Object Model* (VDOM) é um conceito de programação onde uma representação ideal, ou “virtual”, da interface do usuário é mantida em memória e sincronizada com o *Documente Object Model* (DOM) “real” por uma biblioteca como o *ReactDOM*. Esse processo é chamado de reconciliação. Com isso, toda vez que acontece alguma alteração na DOM real, o virtual se modifica imediatamente. (KENZIE, 2022)

### 2.2 JSX

JSX significa Javascript XML. O JSX permite escrever HTML dentro do *React*. O JSX foi criado pela equipe de desenvolvimento do *React*, sendo ele uma forma de criar elementos para serem utilizados como *templates* ou componentes dentro de aplicações feita em *React*. Apesar do JSX ser muito

similar ao HTML, ele não pode ser interpretado pelo navegador. Sendo necessário "transpiladores" como a ferramenta Babel.

O babel é um transpilador *JavaScript*. Quando você tem um código ES2015, que é a versão atual de *JavaScript*, deve utilizar o transpilador para converter o código novo e gerar, assim, um novo código em ES5, que é uma versão antiga do *JavaScript*. O transpilador é necessário como uma ferramenta que lê o código fonte escrito em uma linguagem de programação e produz um código equivalente, com o mesmo nível de abstração (COODSHE, 2021)

O JSX escreve estruturas HTML/XML do mesmo modo que escreve o código *javascript* e o transpilador transforma essas expressões em código *javascript*. Que acontece é que ao invés de colocar *javascript* no HTML, colocamos o HTML no *javascript*. (MUNDOJS, 2022)

Apresentando vantagens, como manter o código mais simples escrevendo grandes pedaços de código. Fornecendo um grande auxílio visual ao se trabalhar com a interface de usuário dentro do código *JavaScript*, também fornece mensagens de erros e aviso pelo próprio *React* que é permitido pelo JSX. Além de ser mais rápido do que o *JavaScript* normal, pois realiza otimizações durante a tradução para o *JavaScript* normal.

## 2.3 Node.js

O *Node.js* é definido como um ambiente de execução *javascript server-side*. Que faz com que aplicações *javascript* sejam criadas para rodar como uma aplicação *standalone* em uma máquina, não dependendo de um browser para execução. (ALURA, 2022)

Com o *Node.js* foi possível diminuir partes da integração da aplicação, como por exemplo a API (*Application Programming Interface*), pois ela é feita direta pelo *Node.js* e consumida pelo *front-end*. O uso de uma mesma linguagem para toda a aplicação também foi uma vantagem bem funcional, sem ter a necessidade de gastar mais tempo aprendendo uma outra linguagem o tempo de desenvolvimento será maior.

## 2.4 Node.js Express

*Node.js Express* é um *framework* descrito como uma biblioteca alternativa para uma série de outros *frameworks* do *Node.js*. Com o *Node.js Express* abre-se a possibilidade de se usar diferentes verbos HTTP, como por exemplo *GET*, *POST* e *DELETE*, dentro da sua aplicação, o que não seria possível apenas com o *Node.js*. Além de gerenciar requisições de diferentes verbos HTTP com *Express* também é possível fazer uma integração com "*view engines*" para inserir dados nos *templates*, definir portas a serem usadas para conexão e a localização dos modelos que são usados para renderizar respostas e adicionar "*middleware*" para novos processos de requisições. Dentro do *Express* é possível usar outros *frameworks*, como o *bcrypt*. (MDN WEB DOCS, 2022)

Pelo fato de se poder usar verbos HTTP dentro da aplicação, facilita muito o uso do *framework*. Devido a esse fato, pode-se gerenciar diferentes requisições URLs, as rotas.

## 2.5 MongoDB

O *mongodb* é um banco de dados com o conceito de ser orientado a documentos. Esses bancos também são chamados de bancos *NoSQL* (*Not Only SQL*). Uma das diferenças desses dois tipos de bancos é que nos orientados a documentos não é fornecido relacionamentos entre documentos, os bancos de dados de documentos ao invés de armazenarem dados relacionados em uma área de armazenamento separado, armazenam no próprio documento. Apenas é possível fazer esse armazenamento separado quando é utilizado uma coleção separada. Sendo a principal diferença dos dois modelos de bancos, é que no modelo relacional se lida com registros, representados por uma abordagem bidimensional (tabelas: linhas e colunas) e no modelo-não-relacional apenas com documentos.

Como é um banco de dados *NoSQL*, ele por si só já é mais flexível e escalável. Com a facilidade de que para adicionar novos dados não é necessário predefini-los no esquema de banco de dado, isso faz com que o processamento fique mais rápido em todos os volumes de dados.

## 2.6 Arquitetura da Aplicação

Com as tecnologias citadas acima, aplicação criada está dentro do modelo cliente-servidor. A aplicação será dividida em duas partes distintas que iram lidar com tarefas específicas, onde o *Node.js* irá lidar com a parte do *back-end*, do lado do servidor, cuidando da camada de dados e provendo informações para as requisições do *frontend*, que quem irá cuidar será o *React*.

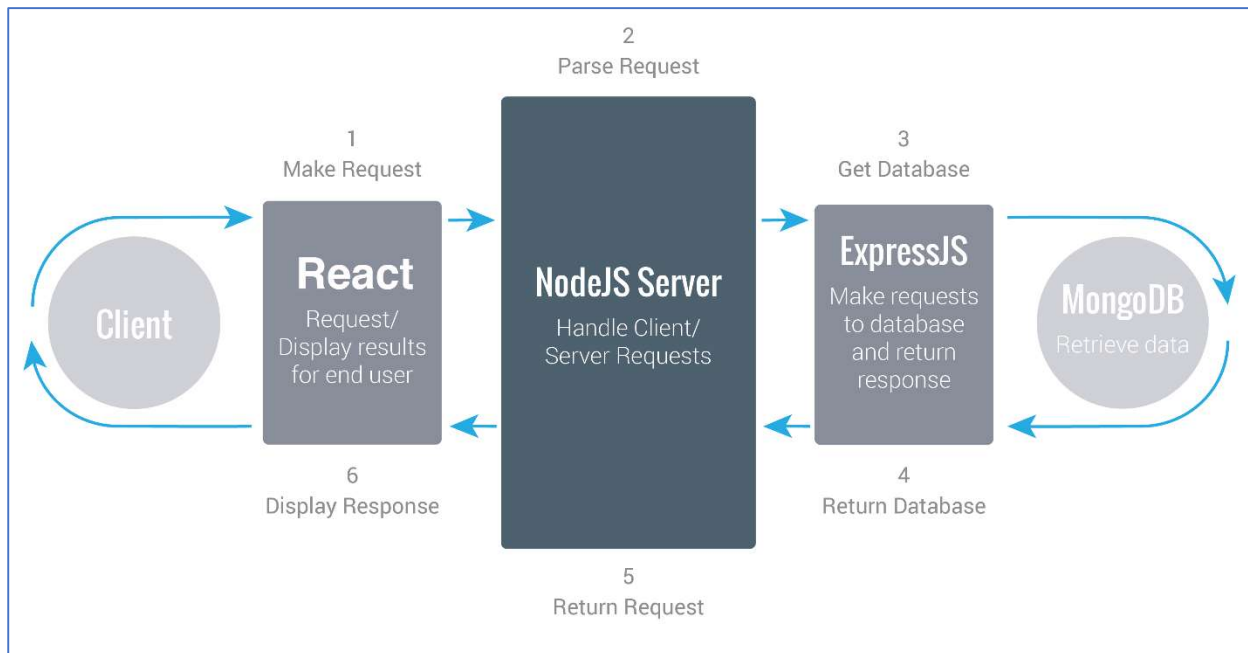
Esse modelo de arquitetura, chamado de Cliente-Servidor, é uma arquitetura onde um programa, o cliente, faz requisições aos serviços de outro programa, o servidor. Existem muitos benefícios em pensar na aplicação em duas partes distintas. O servidor sendo uma entidade com uma API na qual podemos acessar todos os nossos dados. (STACK, 2022)

A Figura 1, apresenta como é o desenho da arquitetura da aplicação. Onde do lado esquerdo o *client* que onde está o *frontend* desenvolvido em *React*, que faz a requisição de serviços, representado pelo número 1 (*make request*) na Figura. Esse pedido de requisição é passado para o servidor, que foi todo desenvolvido em *Node.js*, onde o servidor fará uma análise do pedido e depois o irá passar ao *Express* que cuida das requisições feitas diretas ao banco de dados, se a requisição estiver certa o *Express* ira retornar os dados resgatados do banco de dados ao servidor.

O servidor irá retornar essa resposta ao *front-end* que agora irá apresentar os dados recebidos em tela do lado do cliente.



Figura 1. Representação da Arquitetura da Aplicação



Fonte: (STACK,2022)

## 3 TRABALHOS RELACIONADOS

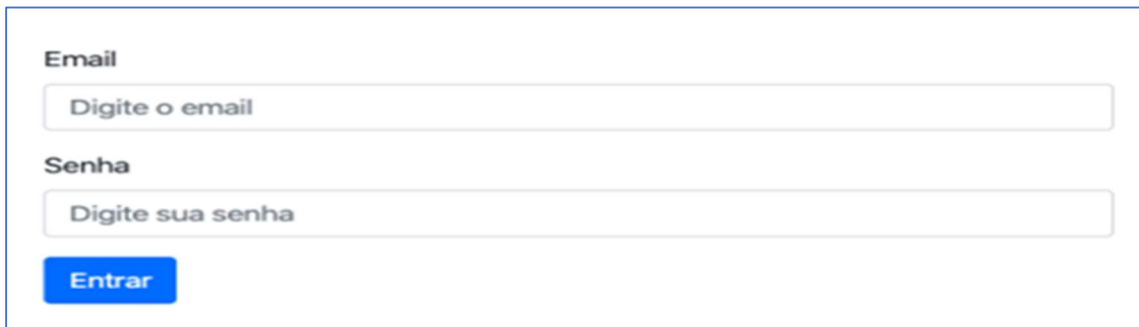
### 3.1 Sistema de Alocação de Salas

O aluno Gabriel de Souza Lins, da Universidade Federal do Ceará em Quixadá, escolheu por desenvolver uma aplicação web com o framework *ReactJS* com *back-end Node.js* que faça alocação de reservas de salas, com o intuito de desenvolver uma ferramenta de apoio aos servidores técnico administrativos e professores da universidade, onde se tenha uma forma mais pratica de alocação e reserva de salas, fazendo que se tenha agilidade nesse processo. (LINS, 2019)

O trabalho selecionado como referência possui relação direta com o projeto de desenvolvimento da aplicação web de avaliar estabelecimentos, pois as ferramentas utilizadas para desenvolvimento de ambas as aplicações propostas foram as mesmas: *ReactJS* e o *Node.js*.

Foi selecionado o *ReactJS* principalmente pela performance que esse framework propõe. Devido ao motivo do *ReactJS* trabalhar com DOM, faz com que a manipulação dos elementos criados para uma tela funcional apresente um ótimo desempenho. Já o *Node.js* foi usado pelo fato de que, assim como o *ReactJS*, possuir como principal vantagem estratégica, o uso de Javascript como linguagem, tornando padrão a forma de desenvolvimento do trabalho, eliminando o tempo extra de esforço. A Figura 2 representa um exemplo da tela de login da aplicação desenvolvida por Gabriel. Dentro da aplicação é feito um controle de acesso a partir do nível de acesso de cada usuário cadastrado no sistema. Na Figura 3 é apresentada a tela inicial da aplicação, onde qualquer usuário que acessar poderá ver as salas que foram reservadas e as que foram alocadas.

Figura 2. Tela de Login



A tela de login apresenta dois campos de entrada de texto e um botão de ação. O primeiro campo, rotulado 'Email', contém o texto 'Digite o email'. O segundo campo, rotulado 'Senha', contém o texto 'Digite sua senha'. Abaixo dos campos, há um botão azul com o texto 'Entrar'.

Fonte: (LINS, 2022).

A Figura 3 é exemplifica a tela inicial da aplicação, evidenciando qual o usuário está logado. Reservas são salas que foram feitas o pedido para o uso, sendo assim classificadas como reservadas. Alocação são as salas que já estão permitidas para o uso, e já passaram pelo pedido de reserva.

Figura 3. Tela Principal



A tela principal é dividida em duas seções principais: 'Reservas' e 'Alocações'. Cada seção contém cartões de informações para diferentes salas.

Reservas	Alocações
<b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Professor(a): Gabriel Disciplina: Célula de Web Dia: 13/12 Horário: 10:0 - 12:0	<b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Dia: Terça Feira Horário: 10:00-12:00
<b>Sala Multiuso 5</b> Bloco 2 Capacidade: 100 Alunos Status: Disponível Professor(a): Admin Disciplina: Seminario Dia: 11/12 Horário: 8:0 - 17:0	<b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Professor(a): Andreia Disciplina: IHC Dia: Segunda Feira Horário: 08:00-10:00

Fonte: (LINS, 2022).

A Figura 4 mostra a tela de salas alocadas e reservadas, onde é possível ver todas as salas cadastradas na aplicação.

Figura 4. Tela de Alocação de Salas

Reservas		Alocações	
<p><b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Professor(a): Gabriel Disciplina: Célula de Web Dia: 13/12 Horário: 10:0 - 12:0</p>	<p><b>Sala Multiuso 5</b> Bloco 2 Capacidade: 100 Alunos Status: Disponível Professor(a): Admin Disciplina: Seminario Dia: 11/12 Horário: 8:0 - 17:0</p>	<p><b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Dia: Terça Feira Horário: 10:00-12:00</p>	<p><b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Professor(a): Andreia Disciplina: IHC Dia: Segunda Feira Horário: 08:00-10:00</p>
<p><b>Sala Multiuso 5</b> Bloco 2 Capacidade: 100 Alunos Status: Disponível Professor(a): Admin Disciplina: seminario Dia: 3/12 Horário: 8:0 - 17:0</p>	<p><b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Professor(a): Gabriel Disciplina: aula extra Dia: 5/12 Horário: 8:0 - 17:0</p>	<p><b>Sala 1</b> Bloco 4 Capacidade: 40 Alunos Status: Disponível Professor(a): Andreia Disciplina: IHC Dia: Terça Feira Horário: 08:00-10:00</p>	

Fonte: (LINS, 2022)

Por fim, justifica-se o uso de *ReactJS* e o *Node.js* para o desenvolvimento da aplicação de avaliação de bares e restaurantes, pois conforme relatado no trabalho proposto o processo de pesquisa e desenvolvimento de um software que atenda uma necessidade de um usuário, tornando-o capaz de realizar suas atividades de forma digital requer um grande esforço para que se obtenha um software que atenda minimamente as expectativas do usuário. (LINS, 2019)

### 3.2 Estudo sobre desenvolvimento de aplicações web com a linguagem javascript e a utilização do framework React

Em Santa Catarina, dentro da Universidade Federal de Santa Catarina, o formando Alexandre Cechinel desenvolveu um estudo sobre como o desenvolvimento web se beneficiaria com o uso de framework e como esses frameworks podem facilitar a vida de um desenvolvedor. (CECHINEL, 2017)

O estudo foi feito sobre *Javascript* e levando em consideração três frameworks: Angular, Vue.js e *ReactJS*. Com a análise desenvolvida, chegou-se à conclusão de que esses frameworks possuem pontos positivos e negativos em relação uns aos outros, porém entregam várias maneiras de ajudar o desenvolvimento web.

O *ReactJS* possui uma flexibilidade e uma grande procura por empresas. Nessa aplicação do estudo foram utilizados vários recursos do *React*, como *JSX* para simplificar a sintaxe do código, o *Hooks* para deixá-lo mais dinâmico, o *Routes* para uma navegação simples entre páginas, e por fim, o uso de vários componentes mais simples para organizar a aplicação. (LOVATTO, 2022)

Com a conclusão de que o *React*, por utilizar o Virtual DOM em vez do DOM real, apresentou um desempenho superior as outras ferramentas estudadas nesse trabalho. Pôr o DOM representar como todos os documentos são organizados em uma estrutura, escolher o *React* para grande manipulação de dados é priorizar o desempenho da aplicação. Angular se tem uma grande curva de aprendizado mesmo não exigindo uma grande quantidade de código pra funcionar. Sendo priorizado quando se pede um tempo de desenvolvimento mais curto.

O próximo capítulo vem com uma explicação detalhada de como é o uso da aplicação de avaliação de bares e restaurantes, considerando os tipos de usuários envolvidos. Serão apresentadas também as funcionalidades de todas as telas da aplicação, com descrições de como o usuário pode se comportar dentro de cada opção.

## 4 AMBIENTE DE PRODUÇÃO E METODOLOGIA

Neste Capítulo, será descrito de forma breve o objetivo, qual foi o ambiente de desenvolvimento utilizado para a produção da aplicação em questão. Em seguida, serão descritas as principais funcionalidades do sistema.

### 4.1 Ambiente de desenvolvimento

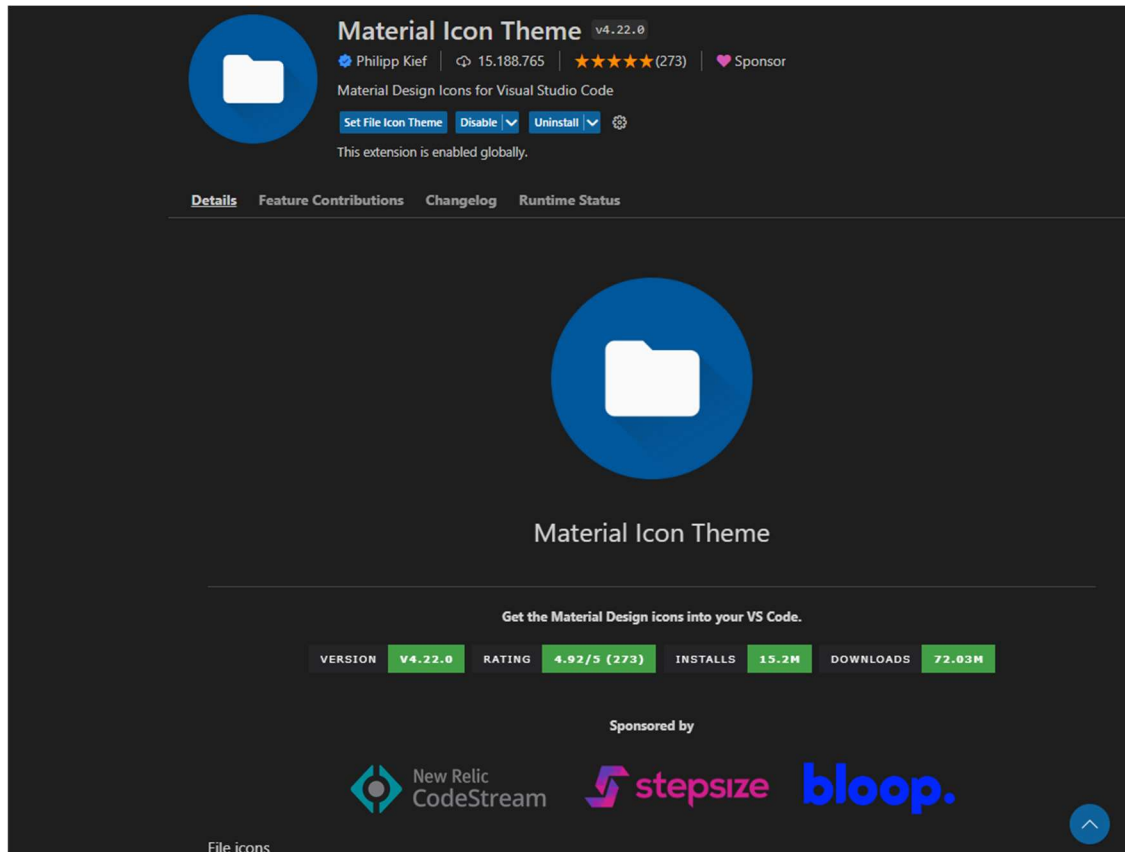
Também conhecida pela sigla inglesa IDE, o Ambiente de Desenvolvimento Integrado é um *software* dotado de várias funcionalidades que facilitam a vida do desenvolvedor.

O *Visual Studio Code* (*VS Code*) é um editor de código-fonte desenvolvido para funcionar em Microsoft para Windows, Linux e MacOS. O suporte a vários tipos de linguagens de programação é um dos diferenciais do *VS Code*. Além disso, a ferramenta permite escolher em qual versão ela será utilizada, apresenta um terminal de comando integrado e também a loja de extensões do ambiente de desenvolvimento.

Esse recurso presente no *VS Code*, a loja de extensão, proporciona com que a ferramenta se torne um editor de código extremamente personalizado. Essas extensões podem ser baixadas direto pelo *VS Code*.

A Figura 5 apresenta a tela com a loja de extensões presente no *VS Code*

Figura 5. Tela VS Code Loja de Extensões



Fonte: do Autor

Esses *plugins* ou extensões auxiliam no desenvolvimento dos projetos, com um aumento de produtividade e agilidade. Dentro do projeto deste trabalho foram utilizadas as extensões *Tabnine AI autocomplete for JavaScript, Python, TypeScript, PHP, Go, Java, Ruby* e demais dependências.

O *Tabnine* foi criado com base em um modelo de inteligência artificial, que usa *Machine Learning* para sugerir o que provavelmente será digitado, em tempo real. Essa extensão fornece uma pequena parte do código-fonte reutilizável (*snippets*), além de reconhecer o código de qualquer linguagem.

Nos próximos tópicos serão descritas todas as funcionalidades presentes no sistema, com imagens ilustrativas pontuando as telas há quais funcionalidades representam.

## 4.2 Descrição da aplicação

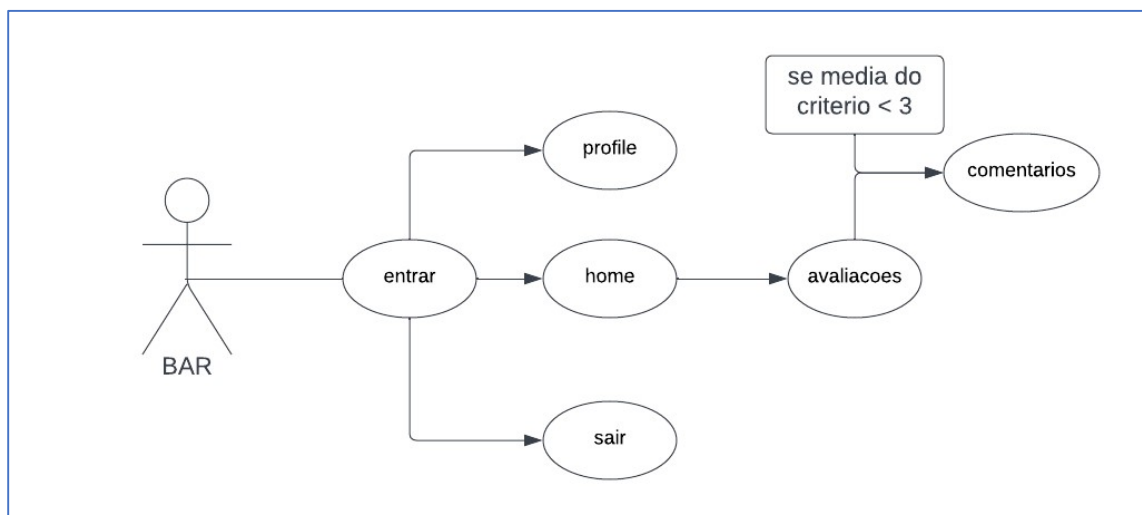
A aplicação desenvolvida neste trabalho teve como foco a criação de um sistema web que possibilite a avaliação de estabelecimentos cadastrados, normalmente pelo seu próprio dono.

A aplicação apresenta a funcionalidade de avaliar vários quesitos desse estabelecimento, como atendimento, se o atendimento foi correto e atendeu todos os pedidos. Ambiente, se o ambiente é agradável, limpo e confortável. Comida, se ela estava bem temperada e de acordo com o que estava descrito no cardápio. E por último preço, se o preço estava dentro da média do mercado.

Esses critérios podem ser avaliados com notas entre 1 e 5, sendo que 1 quer dizer muito ruim e 5 quer dizer muito bom. Portanto, tem-se dois tipos principais de usuário: o cliente (avaliador) e o restaurante/bar (avaliado).

Na Figura 6 é apresentado o caso de uso do usuário bar, onde é mostrado todas as opções possíveis serem realizadas por ele. Ele tem a possibilidade de acessar a aplicação, com ele acessado e autenticado fica disponível para ele a opção de edição de perfil. Também é possível com ele visite a tela de avaliações recebidas, que em caso da média de algum dos critérios for menor que 3, será feito um comentário com recomendações de melhorias.

Figura 6. Caso de Uso Usuário Bar

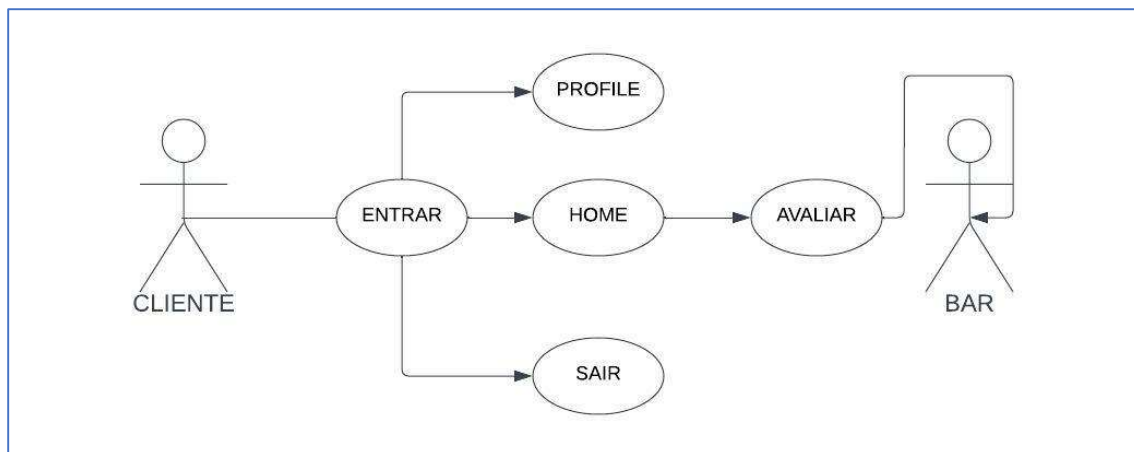


Fonte: do Autor



Na Figura 7, é apresentado o caso de uso do usuário cliente, onde é mostrado que ele pode executar a funcionalidade de edição de perfil. Dentro da página principal (*home*) ele terá a opção de avaliar um estabelecimento (usuário bar)

Figura 7. Caso de Uso Usuário Cliente



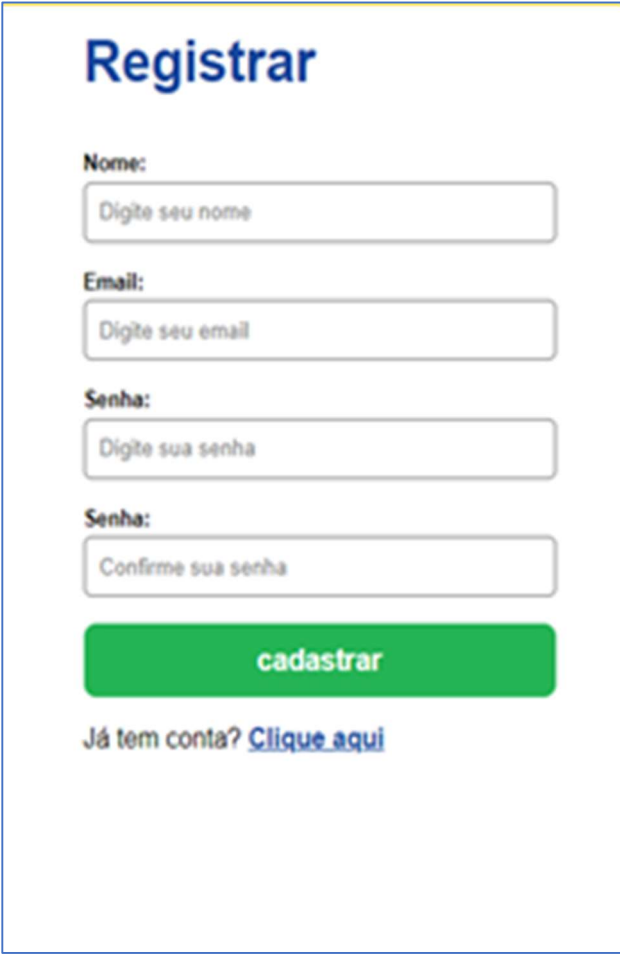
Fonte: do Autor

O usuário cliente além de avaliar estabelecimentos, pode também checar estabelecimentos avaliados, como meio recuperar informações sobre o estabelecimento que deseja frequentar ou visitar.

O usuário denominado cliente, após seu cadastro, terá acesso garantido ao sistema, entrando na tela inicial da aplicação. Nessa tela é exposto ao cliente, todos os estabelecimentos já cadastrados no sistema, com as informações de nome, média do somatório de todas as notas e um botão para avaliação, que apenas é possível clicar se o usuário estiver autenticado. Ao topo da página, é onde está presente a barra de menu (Header), tem-se as opções cadastrar, fazer login ou acessar a página inicial.

Se opção escolhida pelo usuário cliente for fazer o cadastro, ele deve clicar na opção cadastrar, depois será redirecionado para uma nova página, onde após essa escolha ele deve preencher um formulário que possui os campos nome, e-mail, senha e imagem, onde só é possível cadastrar uma imagem de perfil. Com o cadastrado realizado, o usuário é redirecionado para a página inicial da aplicação. A Figura 8 apresenta a tela de cadastro ou registro que pode ser acessada pelo usuário que deseja se cadastrar como um usuário cliente.

Figura 8. Tela de Cadastro Usuário cliente



**Registrar**

**Nome:**

**Email:**

**Senha:**

**Senha:**

**cadastar**

Já tem conta? [Clique aqui](#)

Fonte: do Autor

Quando é escolhido pelo usuário fazer o login, ele deve clicar no botão entrar presente na barra de menu ao topo da aplicação, com isso ele será redirecionado para tela de login, onde o usuário deve preencher o formulário com os dados de acesso, sendo eles e-mail e senha. O usuário será enviado de volta a página principal e agora estará autenticado.

Na Figura 9 é mostrado como é a tela de login.

Figura 9. Tela de Login do Usuário Cliente

# Login

**Email:**

**Senha:**

**Entrar**

Não tem conta? [Clique aqui](#)

Fonte: do Autor

Com o usuário logado e autenticado, a opção de avaliar um estabelecimento fica disponível para ele. A Figura 10 apresenta a tela principal da aplicação com um usuário autenticado.

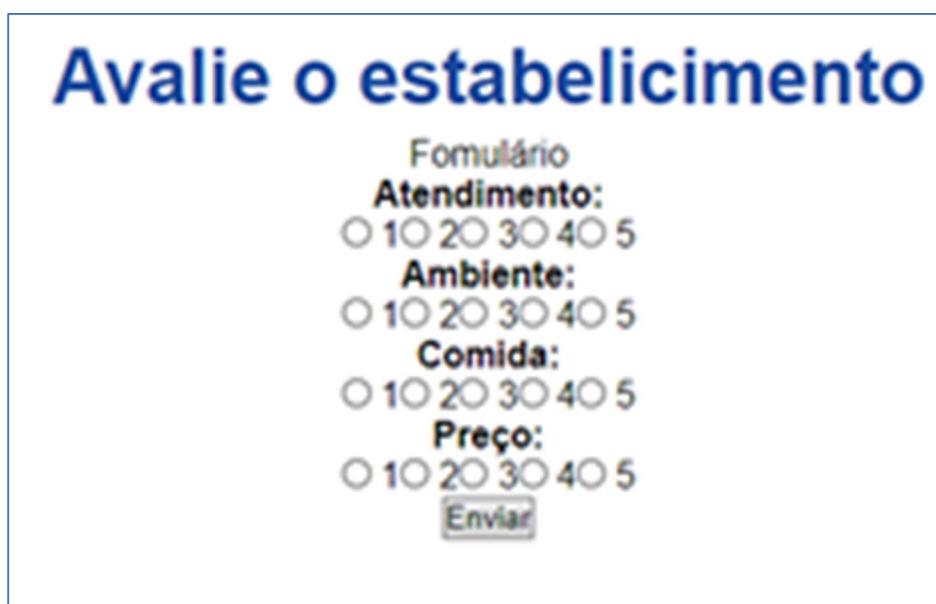
Figura 10. Tela Principal da Aplicação



Fonte: do Autor

Com o usuário autenticado, caso ele desejar avaliar um estabelecimento, ele deve clicar no botão avaliar que fica logo abaixo do estabelecimento escolhido, sendo assim redirecionado à tela de avaliação. Na tela de avaliação, é apresentado um formulário com quatro critérios diferentes para serem avaliados, como atendimento, ambiente, comida e preço, conforme pode ser visto na Figura 11. Os critérios apenas podem ser avaliados com notas de um a cinco, após a avaliação ser realizada ele receberá uma mensagem de que avaliação foi feita com sucesso e será retornado a página principal da aplicação.

Figura 11. Tela de Realizar Avaliação



**Avalie o estabelecimento**

Fomulário

**Atendimento:**  
 1  2  3  4  5

**Ambiente:**  
 1  2  3  4  5

**Comida:**  
 1  2  3  4  5

**Preço:**  
 1  2  3  4  5

Fonte: do Autor

O usuário também possui a opção de edição de perfil, para a realização alguma mudança. Para isso, o usuário deve clicar no botão Profile presente na barra de menu ao topo da aplicação. Na tela de edição de perfil o usuário pode alterar todos os dados cadastrado por ele, como e-mail, senha. Nessa funcionalidade também é possível com que o usuário insira uma imagem de perfil. A Figura 12 apresenta a tela de edição de profile.

Figura 12. Tela de Edição de Perfil

**Perfil**

Imagem:  
Escolher arquivo Nenhum arquivo escolhido

Nome:  
jão

Email:  
joao@email.com

Senha:  
Digite sua senha

Confirmação de senha:  
Confirme sua senha

**Editar**

Fonte: do Autor

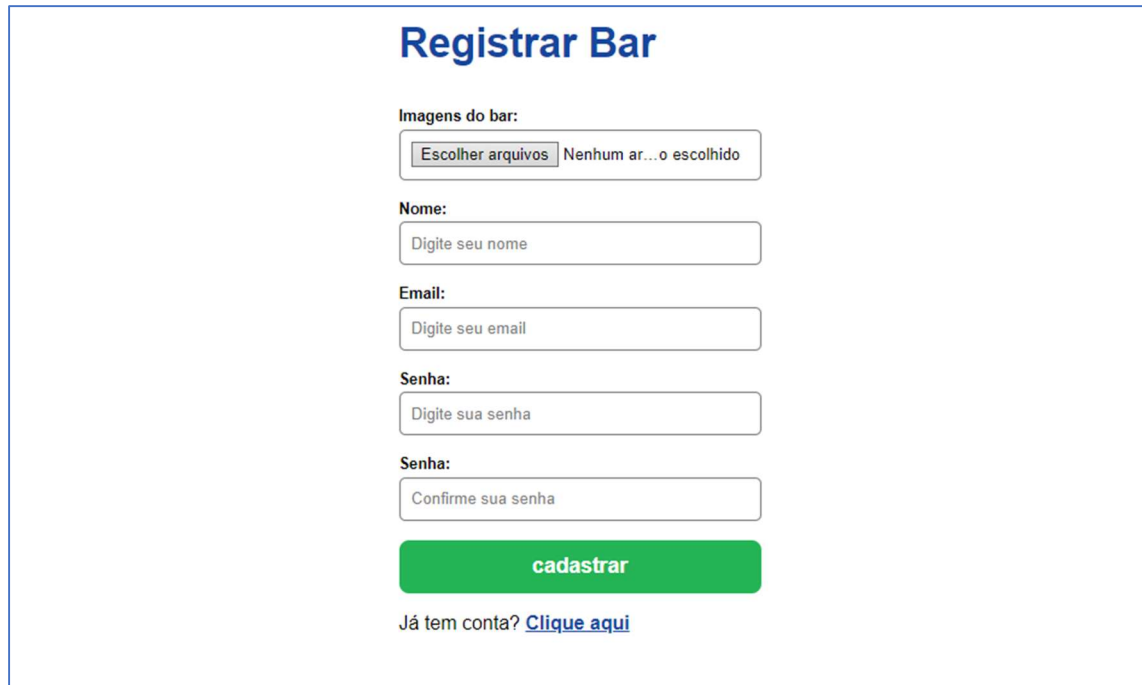
Adicionalmente, na barra de menu, é apresentada a opção sair, que é a opção para usuário realizar o *logout* da aplicação.

O usuário denominado estabelecimento/bar consegue usufruir da funcionalidade de checar as avaliações feitas ao seu estabelecimento e recebe comentários conforme a média de avaliações recebidas. A pontuação calculada para cada estabelecimento leva em consideração, melhorias que devem ser realizadas, caso os itens pontuados estejam com uma média menor que 3 pontos. Essas melhorias são apresentadas aos restaurantes/bares, contendo dicas e instruções para cada item avaliado.

O usuário do estabelecimento/bar apenas poderá ver a página inicial com os estabelecimentos cadastrados quando não estiver logado ao sistema. Se o usuário escolher a opção cadastrar na barra de menu, ele também será redirecionado para página de cadastrado, com o formulário para ser preenchido, possuindo as opções de nome, e-mail, senha e imagens, agora com a

possibilidade incluir quantas imagens for desejada pelo usuário. Como apresentado na Figura 13.

Figura 13. Tela de Cadastro do Usuário Bar



A tela de cadastro do usuário bar, intitulada "Registrar Bar", apresenta os seguintes campos e elementos:

- Imagens do bar:** Um botão "Escolher arquivos" e o texto "Nenhum ar...o escolhido".
- Nome:** Um campo de texto com o placeholder "Digite seu nome".
- Email:** Um campo de texto com o placeholder "Digite seu email".
- Senha:** Um campo de texto com o placeholder "Digite sua senha".
- Senha:** Um campo de texto com o placeholder "Confirme sua senha".
- Um botão verde com o texto "cadastrar".
- Um link azul com o texto "Já tem conta? [Clique aqui](#)".

Fonte: do Autor

Após o cadastrado o usuário do estabelecimento/bar é redirecionado para página de avaliações que ele recebeu. Nessa página é possível ver todas as avaliações recebidas por ele, onde também é apresentado uma média para cada critério de avaliação, os critérios com média 3 ou menor são destacados e com isso é apresentado sugestões de melhorias em forma de texto ao usuário.

A Figura 14 apresenta a tela de avaliações recebidas pelo usuário bar.

Figura 14. Tela de Avaliações Recebidas pelo Usuário Bar

Minhas Avaliações					
Media: 3	Ambiente: 3	Atendimento: 3	Comida: 3	Preço: 3	Avaliado por: cristiano
Media: 2.5	Ambiente: 3	Atendimento: 3	Comida: 1	Preço: 3	Avaliado por: cristiano
Media: 4	Ambiente: 5	Atendimento: 4	Comida: 2	Preço: 5	Avaliado por: cristiano
Media: 4	Ambiente: 5	Atendimento: 4	Comida: 2	Preço: 5	Avaliado por: cristiano

Sugestões
Preço: 3.3333333333333335
Comida: 2 A comida do seu estabelecimento está com média menor que 3, aconselhamos trocar o cozinheiro.
Atendimento: 3.6666666666666665
Ambiente: 3.3333333333333335

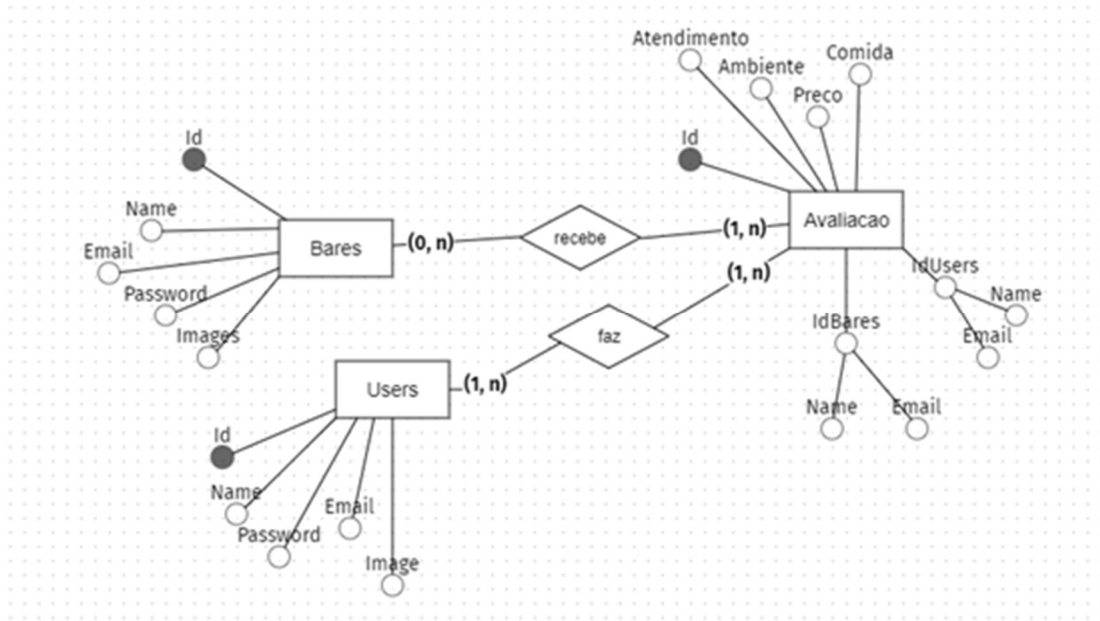
Fonte: do Autor

O usuário do estabelecimento/bar possui apenas 3 opções de escolha em sua barra de menu, sendo elas a página de avaliações recebidas, profile, que é a página onde é possível fazer alteração dos dados cadastrados, como nome, e-mail, senha e imagens. E por último a opção sair, que é a funcionalidade de logout do sistema.

### 4.3 Diagrama do banco de dados

A Figura 15 representa o modelo conceitual do banco de dados usado nesse trabalho. Nela estão presentes as *collections* criadas e as relações que cada uma possui com a outra. A *collection Users* possui os atributos *id*, *name*, *password*, *e-mail* e *image* que tem uma relação 1,n com a *collections Avaliação* que é composta pelos atributos *id*, *atendimento*, *ambiente*, *preço*, *comida*, além dos atributos recebidos das outras *collections* que são: *idUsers*, *nameUser*, *emailUser*, *IdBar*, *nameBar* e *emailBar*. A *collection avaliação* tem uma relação de n,1 com a *collection Bares* que recebe as avaliações criadas pelos *Users*. Na *collection Bares* existem os atributos *id*, *name*, *email*, *password* e *images*.

Figura 15. Modelo Conceitual do Banco



Fonte: do Autor



## 5 RESULTADOS

O *back-end* da aplicação foi desenvolvido em *Node.js*. Através do *Node.js* é possível fazer o envio de instruções sem precisar de um navegador ativo. Logo, foi possível construir toda a parte do *back-end* antes mesmo de começar o *front-end*. O resultado dessa estratégia é que o *front-end* precisou apenas consumir as APIs já criadas em *Node.js*.

*Node.js* possui um ambiente de execução assíncrono. Isso, aliado ao *JavaScript*, fez com que as APIs desenvolvidas para o *back-end* tivessem boa performance, já que o *JavaScript* faz bastante uso de APIs assíncronas.

O NPM (*Node Package Manage*) presente no *Node.js* trouxe com ele os pacotes necessários para que o resultado do desenvolvimento da aplicação acontecesse da melhor maneira, sem a utilização de outros pacotes como por exemplo o *Express*. O resultado da aplicação apresentou-se mais eficiente, principalmente na etapa de construção das rotas com as suas requisições HTTPS.

A ligação entre o *Node.js* e *MongoDB* apresentou resultados satisfatórios, com relação às operações primárias de bancos de dados. Isso ocorreu, pelo fato de que com o *Node.js*, não foi necessário converter os modelos de dados, pois os mesmos objetos *JavaScript* armazenados na base de dados podem ser enviados para o *front-end* sem a necessidade de nenhum tipo de tratamento.

O *front-end* da aplicação foi desenvolvido em *ReactJS* que, por ser essencialmente *JavaScript*, possibilitou o consumo das APIs pelo *front-end* com um bom desempenho, principalmente devido ao uso das APIs assíncronas.

O resultado do uso da “componentização” do *React* foi o grande diferencial. Pelo *React*, foi possível gastar mais tempo otimizando a aplicação do que criando pedaços de códigos repetidos para páginas diferentes. A arquitetura de componentes permitiu com que se fosse usado um componente, como por exemplo os formulários, em páginas diferentes e não necessariamente com as mesmas características. Esse processo foi possível de ser realizado, apenas através da omissão de alguma característica da página, utilizando-se, para isso, uma estrutura condicional simples para retirá-la.

Como tudo em que se envolve tecnologia, há sempre espaços para evolução e, no caso da aplicação web de avaliação de bares/restaurantes, não é diferente. Durante os testes realizados para averiguar como seria a experiência de qualquer tipo de usuário dentro da aplicação chegou-se à conclusão de que ainda existem algumas ações para torná-la melhor, segue:

- na página principal, onde usuário clientes visualizam os estabelecimentos, a criação de filtro para facilitar a pesquisa por um estabelecimento específico;
- criar uma opção de notificação ao usuário bar/restaurante, para que ele saiba quando houver uma nova avaliação ao seu estabelecimento;
- melhorar como é a visualização na página de avaliação feitas ao estabelecimento.

## 6 REFERÊNCIAS

### **5 extensões para melhorar a produtividade no Visual Studio Code.**

**TerraLAB, 2020** Disponível em <http://www2.decom.ufop.br/terralab/5-extensoes-para-melhorar-a-produtividade-no-visual-studio-code/#:~:text=4%C2%BA%20%E2%80%93%20Tabnine,o%20c%C3%B3digo%20de%20qualquer%20linguagem>. Acesso em março de 2022.

### **Brasil é o segundo país do mundo a passar mais tempo na internet.**

**Canaltech, 2022** Disponível em: <https://canaltech.com.br/internet/brasil-e-o-segundo-pais-do-mundo-a-passar-mais-tempo-na-internet-131925/>. Acesso em 5 out, 2022

### **Brasileiros passam mais da metade de suas vidas na Internet, estima pesquisa. Techtudo, 2022.**

Disponível em: <https://www.techtudo.com.br/noticias/2022/05/brasileiros-passam-mais-da-metade-de-suas-vidas-na-internet-estima-pesquisa.ghml>. acesso em outubro de 2022

CECHINEL, ALEXANDRE. **Avaliação do framework Angular e das bibliotecas React e Knockout para o desenvolvimento do Frontend de aplicações Web.** [S. l.] – UFSC, 2017. Disponível em: <https://repositorio.ufsc.br/xmlui/bitstream/handle/123456789/182199/TCC%20PROJETOS%20%20-%20ALEXANDRE%20CECHINEL.pdf?sequence=1&isAllowed=y>. Acesso em: novembro de 2022

### **Conheça as 10 principais linguagens Front-End. Xpeducacao, 2022.**

Disponível em: <https://blog.xpeducacao.com.br/linguagens-front-end/>. Acesso em outubro de 2022

### **Desenvolvimento Web: conceito, vantagens e carreira. Voomp, 2021**

Disponível em: <https://blog.voomp.com.br/graduacao/tecnologia/desenvolvimento-web-conceito-vantagens-e-carreira>. Acesso em outubro de 2022

### **Desenvolvimento Web: Guia Sobre A Criação De Sites E Aplicativos.**

**Mundodevops, 2019** Disponível em <https://mundodevops.com/blog/desenvolvimento-web/>. Acesso em 20 mar, 2022.

### **Introdução ao Visual Studio Code. Devmedia, 2020**

Disponível em <https://www.devmedia.com.br/introducao-ao-visual-studio-code/34418>. Acesso em 20 de mar, 2022.

### **Introdução Express/Node. Mdn Web Docs, 2022**

Disponível em [https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/Express\\_Nodejs/Introduction](https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/Express_Nodejs/Introduction). Acesso em novembro de 2022

**Introduzindo JSX. React, 2022.** Disponível em <https://pt-br.reactjs.org/docs/introducing-jsx.html>. Acesso em outubro de 2022

**Introdução ao MongoDB. Devmedia, 2022.** Disponível em: <https://www.devmedia.com.br/introducao-ao-mongodb/30792>. Acesso em outubro de 2022

**JavaScript. Mdn Web Docs, 2022.** Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>  
Acesso em outubro de 2022

**Node.JS: definição, características, vantagens e usos possíveis. Alura, 2022.** Disponível em: <https://www.alura.com.br/artigos/node-js-definicao-caracteristicas-vantagens-usos>. Acesso em 5 out, 2022

**Node.js – O que é, como funciona e quais as vantagens. Opus Software, 2018.** Disponível em <https://www.opus-software.com.br/node-js/#>. Acesso em outubro de 2022

LINS, GABRIEL. UTILIZANDO REACTJS PARA O DESENVOLVIMENTO DE UM SISTEMA DE ALOCAÇÃO E RESERVA DE SALAS NO CAMPUS DA UFC EM QUIXADÁ. [S. I.] – UFC, 2019. Disponível em: [https://repositorio.ufc.br/bitstream/riufc/49762/1/2019\\_tcc\\_gdeslins.pdf](https://repositorio.ufc.br/bitstream/riufc/49762/1/2019_tcc_gdeslins.pdf). Acesso em: novembro de 2022

**O Que é React e Como Funciona? - Guia para Iniciantes. Hostinger, 2021.** Disponível em <https://www.hostinger.com.br/tutoriais/o-que-e-react-javascript>. Acesso em março de 2022.

**O que é JSX? MundoJS, 2020** Disponível em: <https://www.mundojs.com.br/2020/09/23/o-que-e-jsx/> Acesso em outubro de 2022

**React: o que é, como funciona e porque usar e como aprender. Kenzie, 2022.** Disponível em <https://kenzie.com.br/blog/react/>. Acesso em março de 2022.

**React – Uma biblioteca JavaScript para criar interfaces de usuário. Reactjs, 2022.** Disponível em <https://pt-br.reactjs.org/>. Acesso em março de 2022.

**VS Code - O que é e por que você deve usar? Treinaweb, 2022** Disponível em: [https://www.treinaweb.com.br/blog/vs-code-o-que-e-e-por-que-voce-deve-usar#:~:text=O%20Visual%20Studio%20Code%20\(VS,com%20HTML%2C%20CSS%20e%20JavaScript.](https://www.treinaweb.com.br/blog/vs-code-o-que-e-e-por-que-voce-deve-usar#:~:text=O%20Visual%20Studio%20Code%20(VS,com%20HTML%2C%20CSS%20e%20JavaScript.) acesso em março de 2022.





PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS  
GABINETE DO REITOR

Av. Universitária, 1089 • Setor Universitário  
Caixa Postal 88 • CEP 74605-010  
Goiânia • Goiás • Brasil  
Fone: (62) 3946.1000  
www.pucgoias.edu.br • reitoria@pucgoias.edu.br

## RESOLUÇÃO nº 038/2020 – CEPE

### ANEXO I

#### APÊNDICE ao TCC

#### Termo de autorização de publicação de produção acadêmica

O(A) estudante Critiano Irias Vieira Filho do Curso de Ciência da Computação, matrícula 2016.1.0028.0278-0, telefone: 629222-5260 e-mail cristianoivf@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado **Aplicação web para avaliação de bares e restaurantes**, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 27 de setembro de 2022.

Assinatura do autor: Cristiano Irias Vieira Filho

Nome completo do autor: Cristiano Irias Vieira Filho

Assinatura do professor-orientador: Anibal Santos Jukemura

Nome completo do professor-orientador: Anibal Santos Jukemura