INSTITUTO FEDERAL DE ALAGOAS

CAMPUS MACEIÓ

CURSO BACHARELADO EM SISTEMAS DE INFORMAÇÃO

EMYLLE CHRYSTINNE DA SILVA EUZÉBIO

JARDEL CLEYSON DA SILVA

RANDOM FOOD – SISTEMA DE RECOMENDAÇÃO DE RESTAURANTES

MACEIÓ, AL

2025

EMYLLE CHRYSTINNE DA SILVA EUZÉBIO
JARDEL CLEYSON DA SILVA

RANDOM FOOD – SISTEMA DE RECOMENDAÇÃO DE RESTAURANTES

Trabalho de Conclusão de Curso apresentado ao Curso Bacharelado em Sistemas de Informação do Instituto Federal de Alagoas, campus Maceió, como requisito para a obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Flávio Mota Medeiros
Co-Orientador: Prof. Dr. Fernando Kenji Kamei

Maceió, AL
2025

Dados Internacionais de Catalogação na Publicação
Instituto Federal de Alagoas
***Campus* Maceió**
Biblioteca Benevides Monte

005.1
E91r Euzébio, Emylle Chrystinne da Silva.

Random food [recurso eletrônico] : sistema de recomendação de restaurante / Emylle Chrystinne da Silva Euzébio, Jardel Cleyson da Silva. – Dados eletrônicos (1 arquivo : 1,46 MB). – 2025.

Sistema requerido: Adobe Acrobat Reader.
Modo de acesso: Internet.
Orientação: Prof. Dr. Flávio Mota Medeiros.
Coorientação: Prof. Dr. Fernando Kenji Kamei.
Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) – Instituto Federal de Alagoas, *Campus* Maceió, Maceió, 2025.

1. Sistemas de Informação. 2. Desenvolvimento de software. 3. Randon food – Sistema de recomendação de restaurante. 4. Aplicação web. I. Jardel Cleyson da Silva. II. Título.

Franciane Monick Gomes de França
Bibliotecária – CRB 4/1831

EMYLLE CHRYSTINNE DA SILVA EUZÉBIO

JARDEL CLEYSON DA SILVA

RANDOM FOOD – SISTEMA DE RECOMENDAÇÃO DE RESTAURANTES

Trabalho de Conclusão de Curso apresentado ao Curso Bacharelado em Sistemas de Informação do Instituto Federal de Alagoas, campus Maceió, como requisito para a obtenção do grau de Bacharel em Sistemas de Informação.

Aprovado em: 07/05/2025

BANCA EXAMINADORA

Documento assinado digitalmente
gov.br FLAVIO MOTA MEDEIROS
Data: 02/06/2025 18:06:31-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Flávio Mota Medeiros (Orientador)

Instituição Federal de Alagoas, Campus Maceió - IFAL

Documento assinado digitalmente
gov.br FERNANDO KENJI KAMEI
Data: 03/06/2025 00:41:09-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Fernando Kenji Kamei (Co-Orientador)

Instituição Federal de Alagoas, Campus Maceió - IFAL

Documento assinado digitalmente
gov.br ELVYS ALVES SOARES
Data: 02/06/2025 15:47:34-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Elvys Alves Soares

Instituição Federal de Alagoas, Campus Maceió - IFAL

Documento assinado digitalmente
gov.br EDISON CAMILO DE MORAES JUNIOR
Data: 02/06/2025 14:12:40-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Edison Camilo de Moraes Júnior

Instituição Federal de Alagoas, Campus Maceió - IFAL

AGRADECIMENTOS

Por Emylle Chrystinne

A Deus, por me ouvir mesmo quando as palavras me faltaram e tudo o que restava eram orações entre lágrimas. Por nunca soltar a minha mão, mesmo quando eu mesma quase soltei. Por ser meu refúgio, minha força e minha paz em meio ao caos.

À minha mãe, a mulher mais incrível que já passou por essa vida. Você é a personificação do amor mais puro e incondicional. Teu abraço me curou, tua fé me sustentou, e tua presença nunca deixou faltar nada — nem carinho, nem coragem, nem força. Esse trabalho também é teu, porque cada conquista minha carrega o teu nome.

Aos meus melhores amigos, Jardel e Arthur, que foram minha base, meu alicerce e minha fortaleza durante todos esses anos de faculdade. Sem vocês, eu não teria chegado até aqui.

Arthur, meu irmão que a vida me deu. A gente briga, implica, se desafia — como todo irmão faz. Mas também se ama, se apoia e se entende no olhar. Você esteve do meu lado nos dias mais baixos e nas risadas mais altas, e isso é amor em forma de amizade.

E Jardel, meu melhor amigo pra todas as horas, meu parceiro de vida para tudo e toda hora. A gente cresceu junto nesses anos, viajou junto, chorou, riu, caiu e levantou juntos. Você me conhece mais do que eu mesma, e esteve aqui em absolutamente todos os momentos — inclusive neste, como meu parceiro de TCC. Essa conquista é nossa, mas principalmente, essa amizade é uma das coisas mais preciosas que eu levo pra vida.

Se tem algo que a faculdade me deu de mais valioso, foi a amizade de vocês dois. O conhecimento fica, os títulos vêm, mas ter vivido tudo isso com vocês me transformou profundamente. Eu sou o que sou hoje porque tive a sorte e o privilégio de ter vocês comigo. E por isso, serei eternamente grata.

Obrigada por tudo, de verdade. Vocês todos são minha força e minha casa.

Por Jardel Cleyson

A Deus, por ser minha rocha, minha luz e minha força nos dias mais difíceis e por ter me sustentado até aqui.

Aos meus pais, por cada conselho, por cada gesto de amor, por cada oração e pelo apoio incondicional que me sustentou mesmo quando eu quis parar.

À minha querida irmã, por cada vez que veio me apoiar quando não via saída, cuja presença constante foi um lembrete de que eu não estava sozinho.

A Emy e Arthur, que tornaram a caminhada mais leve. Os dois grandes amigos que Deus me trouxe e que caminharam ao meu lado nos momentos em que tudo parecia ruir — por cada palavra de incentivo, por cada madrugada de estudos, por cada final de semana, por me lembrarem quem eu sou quando até eu esquecia (principalmente enquanto estava naquele outro emprego complicado). Sem vocês, essa conquista não existiria. Vocês são parte indissociável da minha história.

E a todos que, em algum momento, escolheram acreditar em mim — saibam que carrego um pedaço de cada um nesta vitória.

RESUMO

Este trabalho de conclusão de curso apresenta o desenvolvimento do Random Food, um sistema de recomendação de restaurantes que busca facilitar a escolha de refeições por meio da aleatoriedade controlada. A proposta surgiu a partir da observação de uma dor comum entre usuários indecisos sobre onde comer, especialmente diante da ampla oferta de estabelecimentos e aplicativos de delivery. O sistema foi construído utilizando React no frontend, Node.js no backend, Firebase/Firestore como banco de dados, e soluções como Vercel, Render e Railway para hospedagem em nuvem gratuita. O projeto adota práticas de versionamento com Git e GitHub e foi documentado com foco em requisitos funcionais, não funcionais e estrutura de banco de dados. Além disso, foram realizados testes para garantir a estabilidade da aplicação. O Random Food já possui registro oficial de software no INPI, e este trabalho busca não apenas descrever sua construção técnica, mas também apresentar sua relevância como solução criativa, funcional e de baixo custo.

Palavras-chave: Sistema de recomendação; Restaurantes; Aleatoriedade; Aplicação web; Random Food; Desenvolvimento de software; Sistemas de Informação;

ABSTRACT

This final course work presents the development of Random Food, a restaurant recommendation system that seeks to facilitate the choice of meals through controlled randomness. The proposal arose from the observation of a common pain point among users who are indecisive about where to eat, especially given the wide range of establishments and delivery apps. The system was built using React on the frontend, Node.js on the backend, Firebase/Firestore as a database, and solutions such as Vercel, Render and Railway for free cloud hosting. The project adopts versioning practices with Git and GitHub and was documented with a focus on functional and non-functional requirements and database structure. In addition, tests were performed to ensure the stability of the application. Random Food already has an official software registration with INPI, and this work seeks not only to describe its technical construction, but also to present its relevance as a creative, functional and low-cost solution.

Keywords: Recommendation system; Restaurants; Randomness; Web application; Random Food; Software development; Information systems;

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS OU SIGLAS

API	*Application Programming Interface*
CSS	*Cascading Style Sheets*
HTTP	*HyperText Transfer Protocol*
INPI	Instituto Nacional da Propriedade Industrial
JSON	*JavaScript Object Notation*
NOSQL	*Not Only SQL*
NPM	*Node Package Manager*
SQL	*Structured Query Language*

SUMÁRIO

1 INTRODUÇÃO

O processo de tomada de decisão está presente em cada momento da vida humana, da hora em que acordamos até a hora de dormir, seja em decidir a roupa que irá utilizar ou a refeição que irá realizar. Como apontam estudos sobre comportamento do consumidor, a dificuldade em decidir pode gerar fadiga decisória, tornando a experiência menos satisfatória (Schwartz, 2004). Quando se pensa no âmbito culinário, o processo de escolha de um restaurante pode ser um processo extremamente demorado, especialmente diante da imensa oferta de opções.

A tecnologia também está presente para auxiliar nesse processo, buscando torná-los rápidos e eficientes. De acordo com Kahneman (2011), sistemas de recomendação podem reduzir significativamente o esforço cognitivo na tomada de decisões, proporcionando uma experiência mais intuitiva e eficaz.

Nesse contexto, este trabalho apresenta o desenvolvimento da plataforma Random Food que pode ser acessado através do link[1], um software web projetado para otimizar e dinamizar o processo de escolha de restaurantes. O sistema sugere estabelecimentos de forma randômica e/ou personalizada através de filtros, eliminando a sobrecarga da escolha e incentivando novas experiências gastronômicas. Através das funcionalidades como o filtros de busca por categoria, preço médio por pessoa, horário de refeição e localização, e, também, a colaboração dos usuários para enriquecimento da base de dados, o Random Food se destaca como uma solução inovadora para facilitar a decisão de onde comer.

[1] Disponível em: < https://randomfood.com.br > Acesso em: 24 abr. 2025.

2 AVALIAÇÃO E REGISTRO DO SISTEMA

Durante o processo de desenvolvimento do Random Food, realizamos uma avaliação e análise estratégica do mesmo, buscando compreender qual seria o mercado e o seu real potencial.

2.1 ESTUDO DOS CLIENTES

2.1.1 B2C (Público-alvo de consumidores)

O público-alvo do RandomFood é composto por pessoas físicas que enfrentam indecisão na escolha de restaurantes. A faixa etária predominante é de jovens adultos entre 18 e 35 anos, majoritariamente conectados às redes sociais e acostumados a utilizar aplicativos para resolver problemas do cotidiano. Esses clientes possuem hábitos variados, mas uma característica comum é o desejo de praticidade e rapidez na tomada de decisões gastronômicas.

Grande parte desses consumidores é composta por profissionais que trabalham em ambientes urbanos, que valorizam experiências gastronômicas diferenciadas sem investir muito tempo na escolha. A escolaridade média do público é de nível superior ou técnico, e muitos deles moram em áreas urbanas de Maceió, especialmente bairros de classe média e alta, como Jatiúca, Ponta Verde e Mangabeiras.

O comportamento dos clientes indica que a decisão de compra é influenciada por recomendações práticas e personalizadas. Esses consumidores valorizam mais a experiência e a usabilidade do aplicativo do que apenas o preço, sendo atraídos por funcionalidades inovadoras e gamificadas. Eles têm familiaridade com o uso de aplicativos como iFood e Google Maps para decisões relacionadas a refeições, mas estão abertos a novas ferramentas que simplifiquem ainda mais o processo.

O mercado potencial abrange toda a cidade de Maceió, com possibilidade de expansão para outras regiões do estado de Alagoas e futuramente outras capitais. A localização remota da operação e o modelo digital garantem que os clientes possam acessar o serviço de qualquer lugar, aumentando a escalabilidade do negócio.

2.1.2 B2B (Público-alvo de estabelecimentos comerciais)

O segmento B2B é composto por shoppings, praças de alimentação, restaurantes e franquias que buscam aumentar sua visibilidade digital. Esses parceiros podem se beneficiar do RandomFood ao atrair novos consumidores, promover eventos especiais e oferecer descontos exclusivos para usuários do aplicativo. Além disso, o Random Food poderá vir a fornecer insights valiosos sobre os padrões de consumo para os estabelecimentos, o que pode ser um diferencial para melhorar as estratégias de marketing e promoções desses parceiros.

2.2 AVALIAÇÃO ESTRATÉGICA

A matriz F.O.F.A. é um instrumento de análise de negócio simples sua finalidade é detectar pontos fortes e fracos de uma empresa, com o objetivo de torná-la mais eficiente e competitiva. O nome é um acrônimo para Forças, Oportunidades, Fraquezas e Ameaças ou, em inglês, análise SWOT (Strenghts, Weaknesses, Opportunities and Threats). Realizar uma análise F.O.F.A. nos leva a pensar nos aspectos favoráveis e desfavoráveis do negócio.

No processo de criação foi realizada uma análise da matriz F.O.F.A., onde identificou-se como forças do Random Food sua proposta inovadora, interface simples e intuitiva, baixo custo operacional e fácil manutenção. Já sua fraqueza se dava apenas a sua base de dados ainda limitada. Também foram observadas diversas oportunidades, como a expansão do mercado de aplicativos gastronômicos, interesse de jovens adultos por soluções tecnológicas praticas e que economizem tempo, e a possibilidade de parcerias com estabelecimentos locais. A necessidade de constante atualização e volatilidade de preferência de usuários aparece como ameaça.

2.3 ANÁLISE DE MERCADO

O foco do público alvo do Random Food são jovens adultos entre 18 e 35 anos que enfrentam a indecisão na hora de escolher o que comer. O perfil do consumidor concentrado em áreas urbanas, atualmente com destaque para os bairros de classe média e alta de Maceió, com potencial de expansão para outras cidades e estados, são pessoas que estão sempre conectadas e buscam rapidez, praticidade e experiências diferenciadas. Quando se trata do

cenário competitivo, tem como concorrentes indiretos Ifood e Google Maps, que fornecem recomendações de restaurantes na região, porém sem a amplitude de filtros e a característica de sugestão aleatória. A falta de concorrência local direta reforça o potencial de crescimento da plataforma.

2.3.1 Estratégias Promocionais

O modelo de precificação foi definido para equilibrar acessibilidade e geração de receita. A versão gratuita do aplicativo é suficiente para atrair novos usuários e gerar engajamento. Podendo vir a ter uma assinatura mensal direcionada a um público que valoriza a personalização avançada e funcionalidades exclusivas.

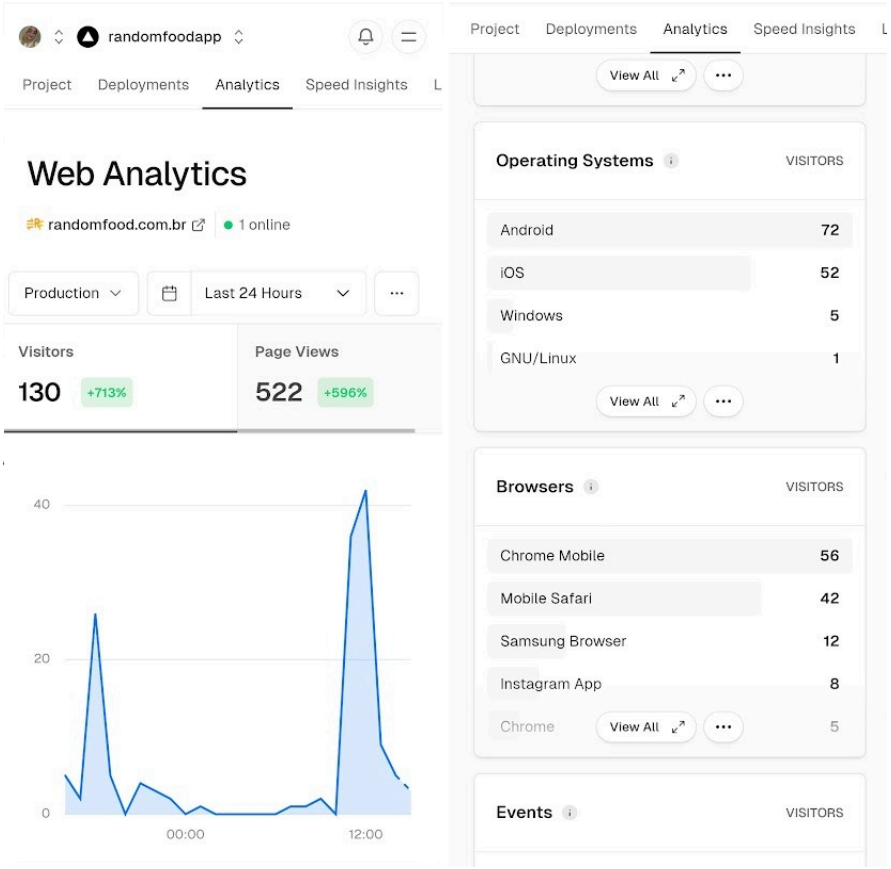
A fim de tornar o software rentável, foi pensando para promover o Random Food campanhas de marketing digital nas redes sociais, especialmente Instagram e TikTok, plataformas populares entre o público-alvo. A estratégia incluirá:

- Parcerias com influenciadores gastronômicos de Maceió, promovendo o aplicativo em postagens e vídeos.
- Anúncios pagos segmentados em plataformas como Facebook Ads e Google Ads, direcionados a moradores de Maceió interessados em experiências gastronômicas.
- Sorteios e promoções para novos usuários, como assinaturas premium gratuitas por um mês.
- Distribuição de material promocional em eventos gastronômicos e tecnológicos locais.
- Além disso, para o público B2B teremos:
- Campanhas direcionadas para restaurantes e shoppings: Promoções em eventos gastronômicos locais e feiras de negócios com foco no setor alimentício.
- Parcerias com influenciadores do setor de alimentação: Utilização de influenciadores gastronômicos e foodies para promover as ofertas especiais dos estabelecimentos parceiros.
- Promoções cruzadas: Oferecimento de descontos para consumidores que visitarem múltiplos estabelecimentos parceiros, incentivando a rotatividade dentro de praças de alimentação e shoppings.

2.4 ANALYTICS

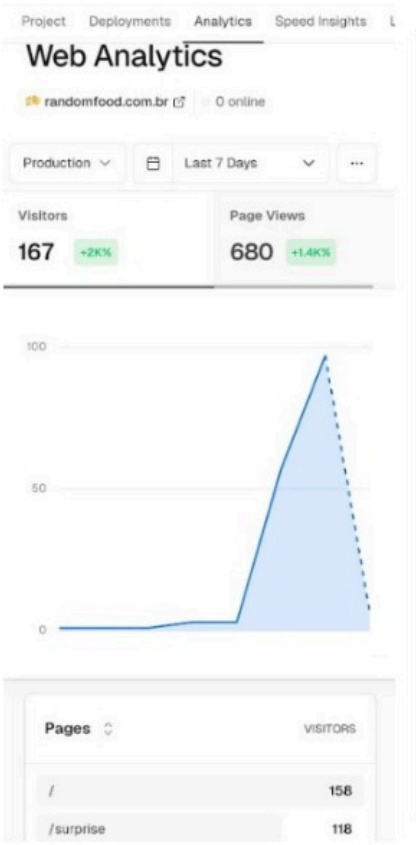
Em agosto de 2024, foi disponibilizada ao público uma versão de teste da plataforma, onde durante um evento de T.I. em Maceió/AL, realizamos testes de usabilidade com os presentes, através da abordagem e apresentação pessoal da plataforma. O evento teve uma duração de dois dias, onde conseguimos coletar os seguintes dados:

Figura 1: Visão 24h - Capturado dia 30 de agosto de 2024



Fonte: Elaborado pelos autores.

Figura 2:Visão 7 dias - Capturado dia 30 de agosto de 2024



Fonte: Elaborado pelos autores.

Estes dados demonstram que até o fim do primeiro dia do evento, tivemos uma média de 167 visitantes na plataforma e de 680 visualizações de páginas, que demonstra que em média cada usuário utilizou 4 vezes a plataforma, testando suas funcionalidades. Além disso, foram recebidos feedbacks positivos sobre a plataforma, validando a proposta, a usabilidade e a interface da mesma.

2.5 REGISTRO DE SOFTWARE

O registro de software é um instrumento essencial para garantir a autoria e proteger os direitos de propriedade intelectual sobre programas de computador. Conforme o artigo 1º da Lei nº 9.609/1998 (Lei do Software), considera-se software a expressão de um conjunto organizado de instruções em linguagem natural ou codificada, capaz de ser interpretado por um sistema computacional para execução de funções específicas. No Brasil, o órgão competente para o registro é o Instituto Nacional da Propriedade Industrial (INPI), autarquia

federal vinculada ao Ministério da Economia, responsável também pela concessão de patentes e registro de marcas. A plataforma Random Food foi devidamente registrada junto ao INPI como Programa de Computador, conforme comprovado no Anexo 1.

3 MÓDULOS DO SISTEMA

3.1 ARQUITETURA DO SISTEMA

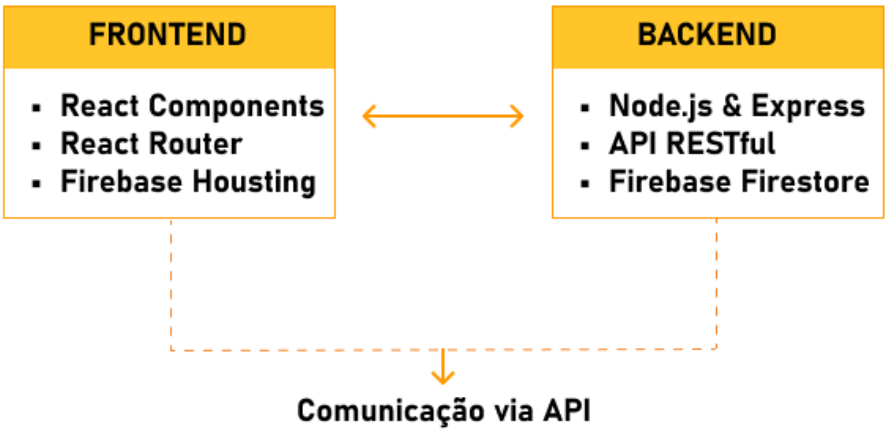
A arquitetura do software foi desenvolvida para ser modular e escalável, utilizando de componentes para facilitar a implantação, manutenção e gerenciamento dos serviços. O sistema é estruturado para garantir a separação clara entre *frontend* e *backend*, permitindo maior flexibilidade na manutenção e evolução da aplicação.

O *frontend* foi desenvolvido utilizando React.js, uma biblioteca JavaScript que facilita a criação de interfaces dinâmicas e componentes reutilizáveis. Para a navegação entre páginas, foi utilizada a biblioteca React Router, que possibilita o roteamento dinâmico de componentes e a atualização da interface sem recarregamento completo da página (SPA – Single Page Application). A estilização da interface foi implementada com CSS, garantindo um layout responsivo e visualmente agradável.

No *backend* foi utilizado Node.js, um ambiente de execução JavaScript que oferece alta escalabilidade e eficiência no processamento de requisições. Para a construção das APIs adotou-se o Express.js, um *framework* minimalista que simplifica a criação de *endpoints* e o gerenciamento de requisições HTTP. O armazenamento de dados é realizado no Firebase Firestore, um banco de dados NoSQL em tempo real, permitindo acesso rápido e eficiente às informações.

A comunicação entre *frontend* e *backend* ocorre por meio de *Application Programming Interface* (APIs), utilizando o protocolo *Hypertext Transfer Protocol* (HTTP) para o envio e recebimento de requisições. O *backend* é responsável por processar os pedidos dos usuários, aplicar a lógica de negócios e consultar o banco de dados para fornecer as respostas apropriadas. Como pode ser observada no diagrama abaixo:

Figura 3 - Arquitetura dos módulos do sistema



Fonte: Elaborado pelos autores.

O fluxo de dados se dá de forma onde o usuário interage com o *frontend* através do navegador, realizando ações como solicitar uma sugestão de restaurante ou aplicar filtros. Em seguida o *frontend* envia requisições via HTTP para o *backend* através das APIs definidas, assim o *backend* processa as requisições, interage com o banco de dados Firestore e aplica a lógica de negócios, armazenando e fornecendo os dados sobre os restaurantes. Recebidos e tratados, o *backend* envia as respostas de volta para o *frontend* que atualiza a interface do usuário com as informações recebidas às apresentando de forma clara.

3.2 PROTOTIPAGEM DA INTERFACE

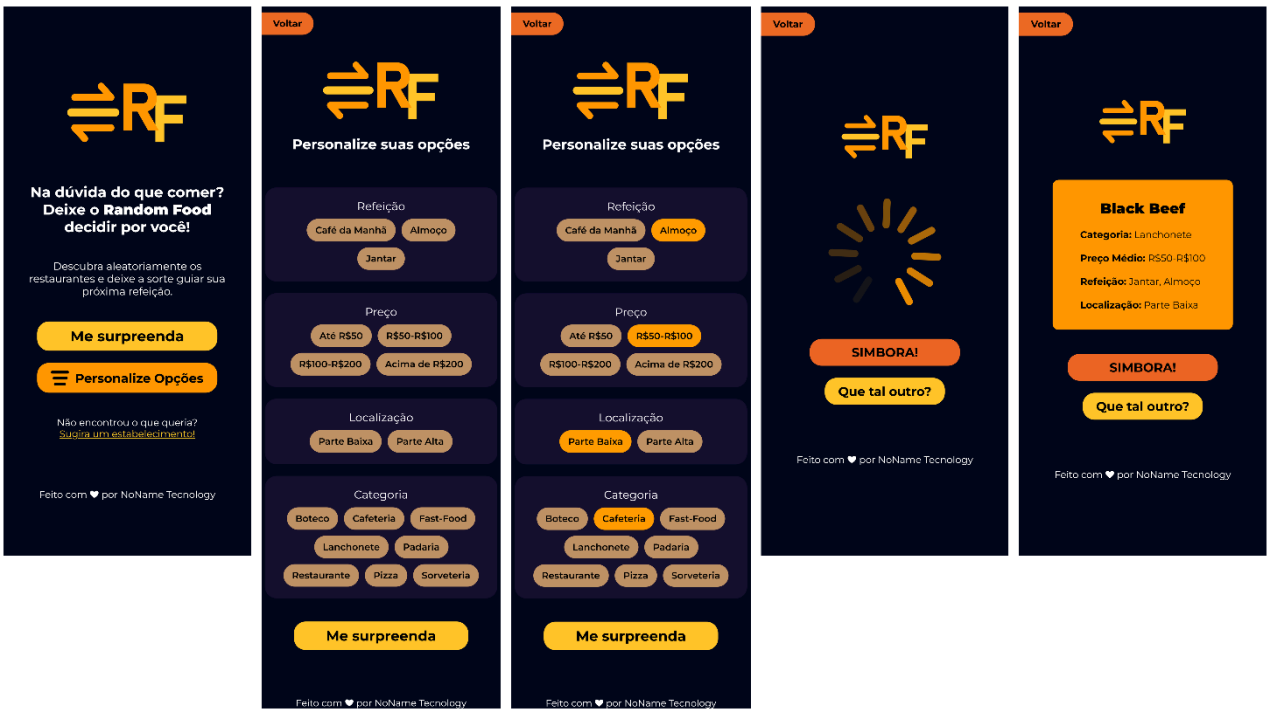
O desenvolvimento da interface foi a primeira etapa de criação da plataforma. Foi criado um protótipo navegável utilizando o Figma (Ferramenta de design vetorial online), buscando compreender a estrutura lógica da plataforma, assim como sua experiência visual. Nele foram incluídas todas as funcionalidades principais do sistema: tela inicial, opções de filtragem e a tela de exibição de resultados.

Desde o começo, a interface foi pensada utilizando o conceito *mobile first*, onde a arquitetura inicial do projeto é pensada para dispositivos móveis, e utilizando os princípios da responsividade, tornando a interface adaptável a qualquer tamanho de tela, garantindo a melhor experiência de uso em todos dispositivos. Esta decisão foi tomada pensando no comportamento atual dos usuários de estarem sempre com seus smartphones, independentemente de onde forem, facilitando o acesso à plataforma na hora de decidir aonde ir. De acordo com a Adobe (2024), projetar com foco inicial em dispositivos móveis é

essencial para garantir uma experiência otimizada desde o início, priorizando clareza, velocidade e funcionalidade em telas menores.

A seguir, apresenta-se o link para acesso ao protótipo navegável desenvolvido no Figma [2], bem como capturas de tela que ilustram algumas das principais telas da aplicação, destacando a abordagem *mobile first* adotada no projeto.

Figura 4 – Telas do Sistema



Fonte: Elaborado pelos autores.

3.2.1. Jornada da Plataforma

Ao acessar a plataforma, na tela inicial, o usuário se depara com as funcionalidades principais: O botão Me Surpreenda e o botão Personalize Opções, além disso também há um link redirecionando para um formulário de sugestão de estabelecimento para ser adicionado a base de dados.

Ao clicar em **Me Surpreenda** o usuário é redirecionado diretamente para tela de resultado, onde é apresentado o estabelecimento recomendado, de forma aleatória, com informações de Nome, Categoria, Preço Médio por Pessoa, Horário de Refeição e Localização. Além disso, também há dois botões, o SIMBORA! que irá redirecionar para a

[2] Disponível em: < https://www.figma.com/design/VoeHECTsCxaj81S3Dr8mrS/Random Food-P%C3%BAblico?node-id=0-1&t=J6gPPvLfByLD21ed-1 > Acesso em: 24 abr. 2025.

localização do estabelecimento no Google Maps e o botão Que Tal Outro? que irá refazer a aleatorização e apresentar um novo resultado.

Já ao clicar em **Personalize Opções**, o usuário é redirecionado para tela de seleção de filtros, onde ele escolherá Categoria, Preço Médio por Pessoa, Horário de Refeição e Localização, precisa escolher no minimo um filtro. Nessa tela, há apenas um botão Me Surpreenda! que redirecionar o usuário para tela de resultado, onde o estabelecimento sugerido é aleatorizado apenas com base naqueles que atendem os filtros.

A jornada do usuário foi desenvolvida com base nas seguintes histórias de usuários:

Tabela 1 – Histórias de Usuários

Como um usuário indeciso, eu quero obter uma sugestão de restaurante para que eu possa decidir rapidamente onde comer.	**Usuário Indeciso** O usuário acessa a página inicial, clica no botão "Me Surpreenda!" e recebe uma sugestão aleatória.
Como um usuário com preferências específicas, eu quero filtrar os restaurantes por tipo de refeição, preço, categoria e localização para que eu possa encontrar opções que atendam às minhas necessidades.	**Usuário com Preferências** O usuário acessa a página inicial, clica no botão "Personalizar", Seleciona os filtros desejados, clica no botão“Me Supreenda”, recebe Resultado.
Como um usuário visitante na cidade, eu quero conseguir encontrar o restaurante sugerido para que eu possa chegar lá mais facilmente.	**Usuário Turista** O usuário acessa a página inicial, clica no botão "Me Surpreenda!" e recebe uma sugestão aleatória, clica no botão “Simbora!” e visualiza a localização do restaurante aleatorizado.

Fonte: Elaborado pelos autores.

3.3 FRONTEND

O *frontend* da plataforma foi desenvolvido utilizando a biblioteca JavaScript, React.js, aliada à estilização realizada com CSS puro. A escolha por essas tecnologias se deu em virtude da capacidade do React em promover a criação de componentes reutilizáveis e altamente personalizáveis, o que contribui para a modularização, manutenção e escalabilidade da interface. A organização do projeto contempla a divisão em pastas temáticas, refletindo uma arquitetura clara e orientada à responsabilidade de cada parte do sistema.

O ambiente de execução utilizado foi o Node.js, o que permite o gerenciamento eficiente do projeto, inclusive no que se refere ao controle de dependências e scripts via npm *(Node Package Manager)*. Este recurso automatiza a instalação, atualização e remoção de bibliotecas e pacotes, otimizando o fluxo de desenvolvimento.

A estrutura de diretórios do *frontend* está organizada da seguinte forma:

- ***public*/**: Contém arquivos estáticos acessíveis diretamente pelo navegador.
- ***src*/**: Diretório principal do código-fonte da aplicação.
    - **assets/**: Armazena imagens e demais recursos estáticos.
    - ***components*/**: Concentra os componentes reutilizáveis desenvolvidos com React.
        - **BackButton**: Permite a navegação para a página anterior.
        - **FilteredOptions**: Apresenta os resultados da filtragem de restaurantes.
        - **FilteringOptions**: Interface para aplicação de filtros por critérios como tipo de refeição, localização, categoria e faixa de preço.
        - **Footer**: Exibe informações complementares no rodapé da página.
        - **Home**: Componente da página inicial.
        - **SurpriseSelection**: Mostra ao usuário uma sugestão aleatória de restaurante com base nos filtros aplicados.
    - ***context*/**: Armazena os contextos utilizados para o gerenciamento de estado global da aplicação.
    - **firebase/**: Contém as configurações de conexão com a plataforma Firebase.

- ***styles***/: Repositório dos arquivos de estilo em CSS.

Para realizar a comunicação entre *frontend* e *backend*, foi utilizada uma API própria, integrada por meio da biblioteca Axios, que facilita o envio de requisições HTTP assíncronas e o tratamento de suas respostas. Essa integração é fundamental para garantir que os dados apresentados na interface sejam atualizados em tempo real e reflitam os filtros e ações realizadas pelos usuários.

3.4 BACKEND

A camada *backend* da aplicação foi construída utilizando o ambiente de execução Node.js, amplamente reconhecido por sua eficiência e escalabilidade em aplicações baseadas em JavaScript. Complementarmente, foi adotado o *framework* Express.js, cuja abordagem minimalista favorece a construção de API REST, ao proporcionar maior controle sobre o roteamento, o tratamento de middlewares e a manipulação das requisições HTTP.

A organização do *backend* segue um modelo modular, com a separação clara entre as camadas de serviço, controle e configuração, o que contribui para a legibilidade, manutenção e expansão do sistema. O armazenamento dos dados é realizado por meio do Cloud Firestore, um banco de dados NoSQL *(Not Only SQL)* gerenciado pelo Firebase, que oferece alta disponibilidade, sincronização em tempo real e estrutura orientada a documentos — características que se alinham às necessidades da aplicação.

O arquivo package.json é utilizado para o gerenciamento das bibliotecas e scripts, garantindo um ambiente controlado e reprodutível. Informações sensíveis, como credenciais de acesso ao banco de dados, são mantidas de forma segura no arquivo .env, impedindo sua exposição em repositórios públicos ou não autorizados.

A seguir, são destacados os principais arquivos e diretórios da estrutura do *backend*:

- **.env:** Armazena variáveis de ambiente confidenciais, como chaves de API e configurações do Firebase.
- **index.js:** Ponto de entrada da aplicação, onde o servidor é inicializado e as rotas principais são carregadas.
- **src/routes/routes.js:** Define os endpoints da API, associando-os às funções responsáveis pelo seu processamento.

- **src/services/restaurantServices.js:** Implementa a lógica de negócio referente ao gerenciamento de dados dos restaurantes, como leitura, filtragem e retorno de informações conforme critérios estabelecidos.

As principais funcionalidades implementadas no *backend* são:

- **Gerenciamento de Restaurantes:**
    - Armazenamento e recuperação de registros em coleções específicas do Firestore.
    - Consulta da lista completa de restaurantes disponíveis.
    - Aplicação de filtros por critérios como categoria, localização, preço e tipo de refeição.
    - Seleção aleatória de um restaurante a partir dos filtros fornecidos.
- **Gerenciamento de Coleções:**
    - Manipulação de outras coleções no Firestore, o que permite a futura ampliação do sistema para incluir diferentes entidades, como usuários, categorias ou avaliações.

As rotas de API disponibilizadas pela aplicação são:

- **GET /api/restaurants**: Retorna todos os registros de restaurantes armazenados.
- **GET /api/restaurants/random**: Seleciona e retorna um restaurante aleatório com base nos filtros fornecidos (ou de forma totalmente aleatória, caso nenhum filtro seja aplicado).
- **GET /api/restaurants/filter**: Permite a busca de restaurantes conforme os parâmetros de consulta, por exemplo: /api/restaurants/filter?categoria=Pizza&localizacao=Parte Alta&preco=R$50-R$100

Essa arquitetura *backend* garante uma base sólida, segura e eficiente para o funcionamento do sistema como um todo, assegurando uma integração harmoniosa com o *frontend* e proporcionando ao usuário uma experiência fluida e confiável.

3.4.1 Fluxo de Dados

Usuário: O usuário interage com o frontend, realizando ações como solicitar uma sugestão de restaurante ou aplicar filtros.

Frontend: Envia requisições HTTP para o backend através das APIs definidas.

Backend: Processa as requisições, interage com o banco de dados Firestore e aplica a lógica de negócios.

Firestore: Armazena e fornece dados sobre os restaurantes.

Backend: Envia as respostas de volta para o frontend.

Frontend: Atualiza a interface do usuário com as informações recebidas e apresenta os resultados para o usuário.

3.4.2 Exemplo de Interação do Usuário

O usuário acessa a página inicial e clica no botão "Me Surpreenda!".

O frontend envia uma requisição GET para o endpoint **/api/restaurants/random**.

O backend consulta o banco de dados Firestore e retorna para um restaurante aleatório.

O frontend exibe as informações do restaurante para o usuário.

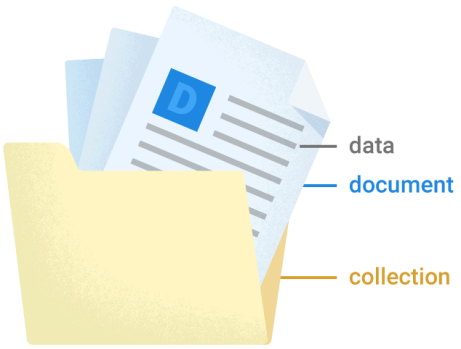
4 BANCO DE DADOS

O sistema Random Food utiliza o Cloud Firestore, serviço de banco de dados NoSQL orientado a documentos fornecido pelo Firebase, como solução para o armazenamento e gerenciamento das informações relativas aos restaurantes cadastrados. Este modelo de banco de dados permite uma estrutura flexível e altamente escalável, sendo ideal para aplicações que exigem respostas em tempo real e manipulação de dados com grande variabilidade.

Diferente de bancos de dados relacionais, o Firestore não utiliza tabelas e relações tradicionais. Em vez disso, organiza os dados em coleções e documentos, onde cada documento pode conter múltiplos campos, inclusive *arrays* e objetos aninhados. Tal estrutura permite maior agilidade na adição de novos campos ou modificações na modelagem dos dados, sem necessidade de reestruturação da base.

A base de dados está estruturada em uma única coleção denominada Random Food, onde cada documento representa um restaurante individual. Essa abordagem favorece a organização lógica dos dados e permite que filtros complexos sejam aplicados de forma eficiente, como por categoria, localização, tipo de refeição e faixa de preço. A Figura 2 ilustra, de forma esquemática, a estrutura da coleção Random Food dentro do Firestore:

Figura 5 - Representação da estrutura de documentos no Firestore



Fonte: Documentação Firestore | Firebase

O acesso ao banco de dados ocorre por meio de chamadas assíncronas definidas no módulo restaurantServices.js, no *backend*. A lógica implementada encapsula a recuperação dos dados, o tratamento das promessas (*promises*) e a formatação dos resultados em *arrays* que são retornados ao *frontend* conforme os critérios de busca definidos pelo usuário.

4.1 CARACTERÍSTICAS TÉCNICAS

- **Nome da coleção:** Random Food
- **Propósito:** Armazenamento de dados dos restaurantes utilizados pela aplicação, como nome, categoria, localização, faixa de preço e tipos de refeições disponíveis.
- **Escalabilidade:** O Firestore permite crescimento horizontal automático, o que facilita o gerenciamento de grandes volumes de documentos e operações simultâneas.
- **Flexibilidade:** A estrutura orientada a documentos permite alterações no esquema dos dados sem comprometer o funcionamento da aplicação, sendo possível adicionar ou remover campos de forma dinâmica.

4.1.1 Componentes dos dados

Nome: Contém o nome do estabelecimento

Categoria: Contém as categorias dos estabelecimentos, podendo ser Boteco, Cafeteria, Fast-Food, Lanchonete, Padaria, Restaurante, Pizza ou Sorveteria

Preço: Contém o preço médio por pessoa, dividido em 4 escalas: Até R$50, R$50-R$100, R$100-R$200, Acima de R$200.

Localização: Contém a localização do estabelecimento dentro de Maceió, dividindo a cidade em 2 partes, sendo do bairro da Gruta de Lourdes em direção ao Aeroporto de Maceió como Parte Alta, e do bairro da Gruta de Lourdes em direção às praias como Parte Baixa.

Refeição: Contém o horário de funcionamento do estabelecimento, divididos em Café da Manhã, Almoço, Jantar.

4.2 ESTRUTURA DOS DOCUMENTOS

Cada documento contido na coleção Random Food corresponde a um restaurante. Os campos utilizados são descritos na Tabela 1:

Tabela 2 - Modo de organização dos documentos no banco de dados

Campo	**Tipo**	**Descrição**	**Exemplo**
nome	string	Nome do Restaurante	“Mc Donald’s”

categoria	string	Categoria do Restaurante	“Fast-Food”
preco	string	Faixa de Preço do Restaurante	“Até R$50”
localizacao	array de strings	Horários de Refeição de Funcionamento do Restaurante	[“Café da Manhã, “Almoço”, “Jantar”]
refeicao	array de strings	Localizações do Restaurante	[“Parte Alta”, “Parte Baixa”]

Fonte: Elaborado pelos autores.

Exemplo de documento:

Bloco de Código em *JavaScript Object Notation* (Json):

```
{
  "nome": "Mc Donald's",
  "categoria": "Fast-Food",
  "preço": "Até R$50",
  "refeicao": ["Café da Manhã", "Almoço", "Jantar"],
  "localizacao": ["Parte Alta", "Parte Baixa"]
}
```

5 AMBIENTE DE HOSPEDAGEM

O sistema Random Food está hospedado integralmente em ambientes de nuvem gratuitos, de modo a viabilizar sua manutenção sem custos adicionais para a equipe de desenvolvimento. O *frontend* está publicado na plataforma Vercel, que permite o *deploy* contínuo de aplicações React com integração facilitada ao Git. Já o *backend* e a API estão distribuídos de forma redundante entre os serviços Railway e Render, garantindo maior disponibilidade e tolerância a falhas.

A base de dados, por sua vez, está estruturada no Firebase Firestore, sendo acessada via requisições HTTP realizadas pelo *backend*, o qual atua como intermediador entre a interface do usuário e o armazenamento em nuvem.

5.1 CONTROLE DE VERSÃO E FLUXO DE DESENVOLVIMENTO

O Random Food adota um fluxo de desenvolvimento simplificado baseado no versionamento de código com Git, por meio da plataforma GitHub, as atualizações e publicações são gerenciadas manualmente por meio de ramificações (*branches*) e requisições de junção (*merge*).

A estratégia consiste na criação de *branches* para o desenvolvimento de novas funcionalidades ou correções, que posteriormente são integradas à *branch* principal (main) após revisão manual e implementação de testes. Esse modelo permite um controle mais direto sobre a evolução do sistema e assegura maior estabilidade nas publicações, especialmente considerando a limitação de recursos presentes no projeto.

6 REQUISITOS

6.1 REQUISITOS FUNCIONAIS

Os requisitos funcionais falam das ações que o sistema oferece. Como o foco principal do Random Food é sugerir restaurantes de forma aleatória, ou também com base nos filtros do usuário, as funções foram pensadas para serem objetivas e fáceis de usar.

Tabela 3 – Requisitos Funcionais

REQUISITOS	DESCRIÇÃO
Sugestão aleatória	Exibe um restaurante aleatório de acordo com os filtros aplicados.
Filtros por preferência	Permite escolher por tipo de refeição, avaliação, distância e faixa de preço.
Integração com banco de dados	Busca os dados no Firebase, que armazena as informações dos restaurantes.
Layout responsivo	Funciona bem em qualquer tela, priorizando dispositivos móveis.
Link para localização	Mostra o local do restaurante no mapa e permite abrir no Google Maps.

Fonte: Elaborado pelos Autores

Cada um dos tópicos foi atendido dentro da plataforma conforme elencado abaixo:

- **Sugestão aleatória:** Opção Me Surpreenda na tela inicial que entrega uma sugestão totalmente sem nenhuma influência do usuário.
- **Filtros por preferência:** Opção Personalize Opções na tela inicial que leva o usuário para uma tela onde o mesmo pode interagir com uma lista de filtros onde o mesmo pode ter uma melhor filtragem daquilo que será sugerido.
- **Integração com banco de dados:** Todas as sugestões são buscadas no banco de dados, sorteadas e entregue ao usuário.

- **Layout responsivo:** O layout foi pensado para se adaptar a todas as resoluções de telas para que a experiência seja a mesma.
- **Link para localização:** Após o resultado da aleatorização ser disponibilizado para o usuário o botão Simbora onde ao selecionar o usuário é levado para o Google Maps e é exibido todas as localizações do respectivo restaurante.

6.2 REQUISITOS NÃO FUNCIONAIS

Aqui entram as qualidades que o sistema precisa ter para funcionar bem e garantir uma boa experiência, mesmo com estrutura gratuita.

Tabela 4 – Requisitos Não Funcionais

REQUISITOS	DESCRIÇÃO
Escalabilidade	*Backend* distribuído entre Railway e Render com redundância, permitindo crescimento futuro.
Segurança	Sem coleta de dados sensíveis. Firebase usado apenas como banco de dados.
Usabilidade	Interface simples e intuitiva, pensada para ser usada com uma mão só, no celular.
Compatibilidade	Funciona nos principais navegadores e sistemas operacionais móveis.
Manutenção	Uso de controle manual por *branches* no GitHub facilita atualizações e correções.
Portabilidade	O sistema pode ser migrado facilmente entre diferentes hospedagens gratuitas e/ou pagas.

Fonte: Elaborado pelos Autores

Os tópicos não funcionais foram alcançados conforme relatado a seguir:

- **Escalabilidade:** O sistema está distribuído em duas plataformas de *Cloud* que juntas trazem maior disponibilidade, segurança e permitem um crescimento futuro.
- **Segurança:** Optamos por não coletar nenhum dado do usuário tanto por questões de privacidade quanto para garantir que em nenhuma das sugestões o sistema tomasse como base nenhuma sugestão anterior, garantindo assim a aleatorização.
- **Compatibilidade:** Toda a plataforma foi pensada para os navegadores mais utilizados atualmente no mercado e que possuem compatibilidade com JavaScript.
- **Manutenção:** Todo o código foi pensado em módulos, onde cada um desses possui suas características e em caso de necessidade de alteração não compromete os demais, garantindo assim clareza e fácil leitura no código.
- **Portabilidade:** Com a modularização do código e tecnologias consolidadas no mercado, a migração entre plataformas caso haja necessidade é facilitada pela escolha das tecnologias que grande parte das estruturas *Clouds* aceitam sem a necessidade de adaptações específicas.

7 TESTES

Neste tópico, descrevemos as ferramentas utilizadas para garantir a qualidade do código e o funcionamento adequado do sistema Random Food. Adotamos ferramentas específicas para realização de testes unitários e análise do código, possibilitando validar o funcionamento e identificar problemas nele.

7.1 TECNOLOGIAS APLICADAS PARA ANÁLISE DE QUALIDADE DO SOFTWARE

Foram utilizadas duas ferramentas principais no processo de verificação da aplicação:

- **Jest**: *Framework* de testes unitários amplamente utilizado em projetos JavaScript e TypeScript. Foi empregado tanto no *frontend* quanto no *backend*, permitindo testar as funcionalidades de forma isolada. Sua utilização contribuiu para assegurar a estabilidade das rotas e da lógica do sistema.
- **Qodana**: Ferramenta de análise estática de código, responsável por identificar inconsistências, possíveis bugs e violações de boas práticas de desenvolvimento. A análise auxiliou na manutenção da legibilidade e qualidade do código, promovendo alinhamento com os princípios de *Clean Code*.

Para realização dos mesmos, criamos diversos casos de testes para as funções, seguindo o molde do exemplo:

- Caso I A função seja chamada corretamente: Retorna todos os restaurantes.
- Caso II A função não receba dados: Retorna o erro 404.
- Caso III Erro de comunicação com o banco de dados: Retorna erro 500.

Testes das Rotas de Restaurantes

I. Teste da Rota GET/restaurants

Função Testada:

```
app.get('/restaurants', async (req, res) => {
  try {
    const data = await getDataFromCollection('RandomFood');
```

```
    if (data.length === 0) {
      return res.status(404).json({ error: 'Não há dados disponíveis na coleção Random Food'
});
    }
    res.status(200).json(data);
  } catch (error) {
    res.status(500).json({ error: 'Erro ao obter dados' });
  }
});
```

Descrição**:** Este conjunto de testes verifica a rota **GET /restaurants**, que deve retornar todos os restaurantes.

Casos de Teste**:**

- I Deve retornar todos os restaurantes:

```
it('should return all restaurants', async () => {
  const mockData = [
    { id: 'Parmegiano', nome: 'Parmegiano' },
    { id: 'Piratas', nome: 'Piratas' }
  ];
  getDataFromCollection.mockResolvedValueOnce(mockData);

  const response = await request(app).get('/restaurants');

  expect(response.status).toBe(200);
  expect(response.body).toEqual(mockData);
});
```

- II Deve retornar 404 se não houver dados:

```
it('should return 404 if there is no data', async () => {
  getDataFromCollection.mockResolvedValueOnce([]);

  const response = await request(app).get('/restaurants');
```

```
  expect(response.status).toBe(404);
  expect(response.body).toEqual({ error: 'Não há dados disponíveis na coleção Random Food' });
});
```

- III Deve retornar 500 em caso de erro:

```
it('should return 500 in case of error', async () => {
  getDataFromCollection.mockRejectedValueOnce(new Error('Erro'));

  const response = await request(app).get('/restaurants');

  expect(response.status).toBe(500);
  expect(response.body).toEqual({ error: 'Erro ao obter dados' });
});
```

7.2 RESULTADO DOS TESTES

Na primeira execução da análise com o Qodana, identificamos 16 problemas classificados entre moderados e críticos, relacionados ao código do *backend*. Com isso, implementámos 8 testes unitários e testes para cada rota, totalizando 11 testes realizados com o Jest.

Após as correções e refatoração nas funções presentes no código, uma nova execução da análise apresentou melhoras, reduzindo para apenas dois alertas críticos, ambos vinculados a dependências externas do React, sem impacto direto na aplicação. Além disso, foi obtida cobertura de testes de 100% nas funcionalidades avaliadas.

8 CONCLUSÃO

O desenvolvimento do Random Food trouxe uma aplicação web funcional e responsiva, utilizando novas tecnologias e uma arquitetura forte capaz de entregar ao usuário uma busca eficiente e personalizada por lugares para comer. A proposta da plataforma vai além da simples aleatoriedade, mas permite aos usuários personalizarem filtros de acordo com a sua preferência, mas ainda mantendo a seleção aleatória.

Durante o processo de construção foram usadas práticas bem estabelecidas do desenvolvimento de software, como o uso de *framework*s comuns no mercado, controle de versão e testes manuais dispositivos e navegadores, para garantir a estabilidade e a usabilidade da aplicação. A implantação da aplicação em ambientes gratuitos na nuvem mostra viabilidade técnica e econômica da solução, tornando a acessível e escalável.

Além do lado tecnológico, o Random Food traz uma grande contribuição social ao ajudar na escolha em situações diárias, como decidir onde comer — algo que pode gerar impasses e um grande gasto de tempo, especialmente em grupos. Ao propor uma única sugestão automática e imparcial, a plataforma colabora para salvar tempo, reduzir indecisões e trazer uma experiência mais tranquila e democrática entre os usuários.

Com os resultados obtidos até o momento, observa-se um notável potencial de expansão e aprimoramento da plataforma. Entre as implementações futuras previstas, destaca-se a criação de um sistema de autenticação por login, que permitirá o salvamento de preferências individuais e o uso contínuo da aplicação com base no perfil de cada usuário. Pretende-se, ainda, introduzir um sistema de avaliação de restaurantes, no qual os próprios usuários poderão atribuir notas e comentar suas experiências, fortalecendo a confiabilidade das recomendações. O cadastro colaborativo de novos estabelecimentos será outra funcionalidade a ser incorporada, permitindo que a base de dados seja constantemente ampliada com sugestões da comunidade. Além disso, será possível configurar preferências alimentares específicas, como alergias, intolerâncias ou dietas restritivas, tornando as sugestões ainda mais precisas e seguras. Por fim, está em curso o planejamento para a expansão geográfica da plataforma, com a inclusão de novas localidades além da área urbana inicial, permitindo que o Random Food seja utilizado em diferentes cidades, bairros e contextos regionais, ampliando seu alcance e impacto social.

REFERÊNCIAS

ADOBE EXPRESS. **Designing mobile-first content**. Adobe, 2024. Disponível em: https://www.adobe.com/uk/express/learn/blog/designing-mobile-first-content. Acesso em: 24 abr. 2025.

BRASIL. **Lei nº 9.609, de 19 de fevereiro de 1998**. Dispõe sobre a proteção da propriedade intelectual de programa de computador. Brasília, DF, 19 fev. 1998. Disponível em: https://www.planalto.gov.br/ccivil_03/leis/l9609.htm. Acesso em: 22 maio 2025.

CSS. **CSS documentation**. Disponível em: https://devdocs.io/css/. Acesso em: 3 abr. 2025.

EXPRESS. **Express routing guide**. Disponível em: https://expressjs.com/en/guide/routing.html. Acesso em: 3 abr. 2025.

FIREBASE. **Documentação do Firebase**. Disponível em: https://firebase.google.com/docs?hl=pt-br. Acesso em: 3 abr. 2025.

INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL (INPI). **Instituto Nacional da Propriedade Industrial**. Disponível em: https://dados.gov.br/dados/organizacoes/visualizar/instituto-nacional-da-propriedade-industrial-inpi#:~:text=Criado%20em%201970%2C%20o%20Instituto,propriedade%20intelectual%20para%20a%20ind%C3%BAstria. Acesso em: 22 maio 2025.

JAVASCRIPT. **JavaScript documentation**. Disponível em: https://developer.mozilla.org/en-US/docs/Web/JavaScript. Acesso em: 3 abr. 2025.

JEST. **Jest documentation**. Disponível em: https://jestjs.io/pt-BR/docs/getting-started. Acesso em: 3 abr. 2025.

KAHNEMAN, Daniel. **Rápido e devagar**: duas formas de pensar. Tradução de Cássio de Arantes Leite. Rio de Janeiro: Objetiva, 2012. E-book.

NODE.JS. **Node.js documentation**. Disponível em: https://nodejs.org/docs/latest/api/documentation.html. Acesso em: 3 abr. 2025.

QODANA. **About Qodana**. Disponível em: https://www.jetbrains.com/help/qodana/about-qodana.html. Acesso em: 3 abr. 2025.

REACT. **React documentation**. Disponível em: https://react.dev/reference/react. Acesso em: 3 abr. 2025.

REACT ROUTER. **React Router documentation**. Disponível em: https://reactrouter.com/home. Acesso em: 3 abr. 2025.

SCHWARTZ, Barry. **The paradox of choice**: why more is less. New York: HarperCollins e-books, 2004. E-book.

ANEXOS

REPÚBLICA FEDERATIVA DO BRASIL
MINISTÉRIO DO DESENVOLVIMENTO, INDÚSTRIA, COMÉRCIO E SERVIÇOS
INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL
DIRETORIA DE PATENTES, PROGRAMAS DE COMPUTADOR E TOPOGRAFIAS DE CIRCUITOS

Certificado de Registro de Programa de Computador

Processo Nº: **BR512025000985-2**

O Instituto Nacional da Propriedade Industrial expede o presente certificado de registro de programa de computador, válido por 50 anos a partir de 1° de janeiro subsequente à data de 03/06/2024, em conformidade com o §2°, art. 2° da Lei 9.609, de 19 de Fevereiro de 1998.

Título: Random Food

Data de publicação: 03/06/2024

Data de criação: 03/06/2024

Titular(es): INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE ALAGOAS - IFAL

Autor(es): FLÁVIO MOTA MEDEIROS; EMYLLE CHRYSTINNE DA SILVA EUZÉBIO; ARTHUR PASSOS DE MELO; JARDEL CLEYSON DA SILVA

Linguagem: NODEJS; OUTROS

Campo de aplicação: IF-10

Tipo de programa: AP-01

Algoritmo hash: SHA-512

Resumo digital hash:
f2d511fb9dc084eb94ffd5b6c9a32f0344172c27d3d5d3002c8d0d414bc5b5622b45047843c8fad38a3ec4eb5dba1eceafafe2a78f28a7d0e578a714e88a224e

Expedido em: 25/03/2025

Aprovado por:
Carlos Alexandre Fernandes Silva
Chefe da DIPTO