Documento de Especificação de Requisitos do Sistema ACESSAÊ

Documento de Especificação de Requisitos do Sistema ACESSAÊ

Grupo: Pedro Judah G. Nogueira Lopes,
Eduardo de F. Santana,
Felipe M. Ventura,
Otávio M. Santos,
Ricardo B. da Silva e
Warmeson de F. Alencar.

Sumário

Introdução

Este documento apresenta a definição dos requisitos funcionais que fundamentarão não só o desenvolvimento do MVP do ACESSAÊ, mas também a metodologia de avaliação utilizada para atribuir a nota aos usuários.

1. Definição dos requisitos de usuário

1.1. Requisitos Funcionais

RF01. Cadastro de Novos Usuários
Informações: Permitir que uma pessoa crie uma nova conta no aplicativo.
Regras:

1. O formulário deve pedir Nome, E-mail e Senha.
2. O sistema deve validar se o e-mail já existe.
3. O sistema deve exigir uma senha com padrão de segurança.

RF02. Autenticação de Usuário
Informações: Permitir que usuários entrem e saiam do app de forma segura.
Regras:

1. A tela de login deve solicitar E-mail e Senha.
2. O sistema deve exibir uma mensagem de erro para dados incorretos.
3. O sistema deve ter uma opção "Esqueci minha senha" que envia um link para o e-mail do usuário.

RF03. Gerenciamento de Usuários (Admin)
Informações: Dar ao administrador uma tela para gerenciar todas as contas.
Regras:

1. O administrador deve ter acesso a uma tela que lista todos os usuários (nome, e-mail, status).
2. O sistema deve permitir a busca por usuários.
3. O administrador deve poder editar e excluir um usuário (com confirmação).

RF04. Gerenciamento de Perfil Pessoal
Informações: Permitir que o usuário edite suas próprias informações.
Regras:

1. O usuário deve ter acesso à página "Meu Perfil".
2. Nesta página, o usuário deve poder editar seu nome.

RF05. Visualização de Locais no Mapa
Informações: Mostrar um mapa interativo com os locais cadastrados.
Regras:

1. O sistema deve utilizar uma API de mapa.
2. O mapa deve, por padrão, focar na localização do usuário.
3. Os locais devem aparecer como "pins" clicáveis no mapa.
4. Ao clicar em um "pin", o sistema deve abrir a página de detalhes do local (RF07).

RF06. Busca de Locais
Informações: Permitir que o usuário pesquise por locais específicos.
Regras:

1. A interface deve conter um campo de busca visível.
2. O sistema deve permitir a busca por nome do estabelecimento.
3. A busca deve mostrar os resultados em uma lista.

RF07. Página de Detalhes do Local
Informações: Apresentar uma página completa com todas as informações de um local.
Regras:

1. A página deve mostrar nome, endereço, telefone (se houver na API) e fotos do lugar.
2. A página deve ter uma seção clara e dedicada para as avaliações de acessibilidade (RF08 e RF09).

RF08. Média de Avaliações
Informações: Mostrar um resumo da acessibilidade do local.
Regras:

1. O sistema deve calcular e mostrar a média de estrelas para cada critério (definidos pelo RF10).
2. O sistema deve exibir uma nota média geral em destaque.

RF09. Listagem de Avaliações Detalhadas
Informações: Mostrar a lista com todas as avaliações que outros usuários fizeram.
Regras:

1. Cada avaliação na lista deve mostrar o nome do usuário, data e as estrelas.
2. O comentário em texto completo deve ser exibido.
3. As fotos enviadas na avaliação devem ser exibidas.
4. O sistema deve permitir ordenar as avaliações por "Mais Recentes" ou "Mais Relevantes".

RF10. Criar Avaliação com Fotos e Critérios
Informações: Permitir que o usuário envie sua própria avaliação completa.
Regras:

1. O formulário deve ter critérios fixos (Acesso, Banheiro, etc.).
2. O usuário deve dar uma nota de 1 a 5 estrelas para cada critério.
3. O formulário deve ter um campo de texto obrigatório para descrever a experiência, com limitação de caracteres.
4. O sistema deve permitir o upload de fotos.
5. O usuário só pode avaliar um local uma vez, mas pode editar sua avaliação.

RF11. Funcionalidades do Perfil (Histórico)
Informações: Mostrar ao usuário um histórico de todas as suas contribuições.
Regras:

1. O perfil do usuário deve ter uma aba "Minhas Avaliações".
2. Esta aba deve listar todos os locais que o usuário já avaliou.
3. A partir desta lista, o usuário deve poder editar ou excluir suas próprias avaliações.

2. Tecnologias e Ferramentas Adotadas

Optou-se por um ecossistema de ferramentas modernas, com foco em produtividade, escalabilidade e redução de custos operacionais, permitindo que a equipe se concentrasse na regra de negócio e na experiência do usuário.

A seguir, detalhamos a stack tecnológica escolhida para cada camada da aplicação.

2.1. Design e Prototipação

Figma: Ferramenta líder de mercado para design de interfaces (UI) e experiência do usuário (UX).

- Justificativa**:** A utilização do Figma na fase inicial foi crucial para desenhar, prototipar e validar todas as telas e fluxos de navegação antes do início do desenvolvimento. Essa abordagem previne retrabalho, alinha a visão da equipe e serve como uma documentação visual clara do produto a ser construído.

2.2. Frontend (Aplicação Web)

Next.js: Um framework full-stack para a construção de aplicações web modernas.

- Justificativa: A escolha do Next.js se deu por sua capacidade de acelerar o desenvolvimento com funcionalidades como roteamento de páginas simplificado, otimização de performance automática e a habilidade de renderizar conteúdo tanto

no servidor quanto no cliente. Isso nos permitiu criar uma aplicação web rápida, responsiva e com uma excelente experiência de usuário.

2.3. Backend as a Service (BaaS)

Google Firebase: Uma plataforma completa que oferece um conjunto de serviços de backend prontos para uso.

- Justificativa: A adoção do Firebase foi uma decisão estratégica para eliminar a necessidade de construir e manter uma infraestrutura de backend do zero. Utilizamos os seguintes serviços:
  - *Firebase Authentication:* Para implementar um sistema de cadastro e login de usuários seguro e completo, com um esforço de desenvolvimento mínimo.
  - *Firestore:* Como nosso banco de dados NoSQL, oferecendo flexibilidade para armazenar os dados de locais e avaliações, além de ser escalável e de fácil integração.
  - *Firebase Storage:* Para gerenciar o upload, armazenamento e entrega das fotos enviadas pelos usuários de forma simples e segura.

2.4. API de Mapas

Leaflet.js e OpenStreetMap: Combinação de uma biblioteca JavaScript de código aberto para mapas interativos (Leaflet) com um provedor de dados de mapa colaborativo e gratuito (OpenStreetMap).

- Justificativa: Optamos por esta solução por ser 100% gratuita e de código aberto, eliminando a burocracia e os potenciais custos associados a APIs comerciais, como a do Google Maps. O Leaflet é leve, poderoso e atende a todos os requisitos funcionais do projeto, como a exibição de mapas e a customização de marcadores.

2.5. Hospedagem e Implantação (Deployment)

Vercel: Plataforma de nuvem otimizada para a hospedagem de aplicações Frontend e, em especial, projetos Next.js.

- Justificativa: A Vercel foi escolhida por sua integração nativa e perfeita com o Next.js. Ela oferece um processo de implantação contínua (CI/CD) automatizado, onde cada atualização no repositório do Git é publicada instantaneamente, agilizando os testes e a entrega de novas funcionalidades. Seu plano gratuito é robusto e ideal para o escopo do projeto.

MER Conceitual do Projeto