**1. Visão Geral da Arquitetura**

* **Frontend**: Interface visual interativa para os usuários (ONGs, beneficiários e administradores do sistema).
* **Backend**: Processamento de lógica de negócio, APIs REST para comunicação e integração com o banco de dados.
* **Banco de Dados**: Armazenamento e gerenciamento de dados, como ONGs cadastradas, beneficiários e doações.

**2. Camada de Frontend**

**Objetivo**: Criar uma interface intuitiva e responsiva para os usuários.

* **Linguagens e Frameworks**:
  + HTML5, CSS3, JavaScript.
  + **Frameworks de Frontend**: Bootstrap para responsividade, React (ou outro framework JavaScript).
  + **Ferramentas de Design**: Figma e Canva para o design e protótipos.
* **Componentes Principais**:
  + **Landing Page**: Apresentação do projeto, informações sobre desperdício alimentar e segurança alimentar.
  + **Formulário de Cadastro para ONGs**: Com campos como nome, endereço, estado, horário de funcionamento, CNPJ, e-mail e telefone.
  + **Sistema de Autenticação**: Login e cadastro de usuários (ONGs e beneficiários).
  + **Listagem de ONGs e Doações**: Exibição das ONGs cadastradas com possibilidade de agendamento de doações.
  + **Seção de Agendamento**: Permite que beneficiários escolham ONGs e horários disponíveis para retirada de alimentos.

**3. Camada de Backend**

**Objetivo**: Processamento das regras de negócio e comunicação com o banco de dados.

* **Tecnologias**:
  + **Node.js e Express** para o desenvolvimento do servidor e APIs REST.
  + **ASP.NET Core** para a camada de backend em C#, caso necessário.
* **Arquitetura**:
  + **Estrutura MVC** (Model-View-Controller) para organização do código.
  + **APIs REST** para conectar o frontend com o backend, incluindo as seguintes rotas:
    - /ongs para gerenciar cadastro, atualização e remoção de ONGs.
    - /doacoes para gerenciar o agendamento e histórico de doações.
    - /users para operações de autenticação, como login, cadastro e recuperação de senha.
* **Principais Funcionalidades**:
  + **Autenticação e Autorização**: Usando JWT (JSON Web Token) para autenticação.
  + **Cadastro e Gestão de ONGs**: CRUD (Create, Read, Update, Delete) para ONGs e doações.
  + **Agendamento de Doações**: Lógica para agendar retirada de alimentos de acordo com o horário de funcionamento das ONGs.
  + **Envio de Notificações**: Notificar ONGs e beneficiários sobre agendamentos ou alterações (opcional).

**4. Camada de Banco de Dados**

**Objetivo**: Armazenamento e gerenciamento de dados de forma segura e eficiente.

* **Tecnologia**:
  + **MySQL** para estrutura de dados relacional.
* **Estrutura de Tabelas**:
  + **ONGs**: Armazena informações das ONGs (nome, endereço, estado, horário de funcionamento, CNPJ, e-mail, telefone).
  + **Beneficiários**: Dados dos beneficiários, caso haja uma necessidade de autenticação.
  + **Doações**: Detalhes de cada doação, incluindo ONG, data de agendamento, tipo de alimento e status da retirada.
  + **Usuários**: Controle de usuários para acesso ao sistema, incluindo autenticação e gerenciamento de permissões.

**5. Integração e Testes**

* **Integração de APIs**: O backend deve estar bem documentado para que o frontend se comunique adequadamente com as APIs.
* **Testes Automatizados**: Implementar testes de unidade e testes de integração para garantir a funcionalidade do sistema.
* **Hospedagem e Deploy**:
  + Frontend: Hospedagem em um serviço como Netlify ou Vercel.
  + Backend: Implementação do servidor em plataformas como Heroku ou DigitalOcean.
  + Banco de Dados: MySQL em uma plataforma de banco de dados em nuvem, como AWS RDS ou Azure SQL Database.

**6. Segurança**

* **Validação e Sanitização de Dados**: Garantir que todos os dados sejam validados e sanitizados para prevenir ataques como SQL Injection.
* **Autenticação Segura**: Uso de HTTPS para todas as requisições, e proteção de endpoints críticos com autenticação JWT.