Algoritmos e Estrutura de dados:

Definir qual estrutura de dados será usada para receber uma lista de dados do servidor:

No lado do servidor (C#):

No backend, uma estrutura de dados recomendada para receber e manipular uma lista de dados no **C#** é uma **List<T>**. Essa é uma estrutura de dados genérica que permite armazenar e gerenciar uma coleção de objetos de forma eficiente.

No lado do cliente (JavaScript/JSON):

Quando os dados são enviados ou recebidos do servidor para o cliente, eles geralmente são transmitidos em **formato JSON**, que é leve e fácil de manipular. No lado do cliente, você pode utilizar uma estrutura de dados como um **array** em **JavaScript** para lidar com a lista de dados recebida.

Implementar um algoritmo de ordenação na lista recebida:

Algoritmo de ordenaçao.cs

Full Stack:

Home do site em HTML, CSS E JS: https://github.com/rafaelacoelhob/Home-da-pagina

Design de Interface digital:

Criação do Layout e Prototipação no FIGMA da Home do Projeto, Definição da paleta de cores do projeto:
https://www.figma.com/design/j4pC2lMTqTk5AlOf6fuSlP/Untitled?node-id=1-3

Modelagem de Software:

Identificar 6 requisitos funcionais e 6 não funcionais do sistema. Construir 3 casos de uso:

Requisitos Funcionais (RF):

1. **Cadastro de doadores e beneficiários** – O sistema deve permitir que usuários se cadastrem como doadores ou beneficiários.
2. **Registro de doações** – O sistema deve registrar as doações realizadas pelos doadores, incluindo tipo de alimento, quantidade e data de validade.
3. **Consulta de doações disponíveis** – Beneficiários devem poder consultar as doações disponíveis, filtrando por tipo de alimento e local.
4. **Agendamento de retirada** – O sistema deve permitir que beneficiários agendem a retirada das doações.
5. **Notificações de novas doações** – O sistema deve notificar os beneficiários quando houver novas doações disponíveis que se encaixem em seus filtros de interesse.
6. **Relatórios de doações e beneficiários** – O sistema deve gerar relatórios de doações feitas e doações retiradas, agrupando por período, tipo de alimento e beneficiário.

Requisitos Não Funcionais (RNF):

1. **Segurança** – O sistema deve garantir a segurança dos dados dos usuários, incluindo informações pessoais e dados de doações.
2. **Desempenho** – O tempo de resposta para consultas de doações não deve exceder 3 segundos em uma base de dados com até 100 mil registros.
3. **Escalabilidade** – O sistema deve ser capaz de lidar com o aumento progressivo de usuários e doações sem perda de desempenho.
4. **Compatibilidade** – O sistema deve ser responsivo e acessível em diferentes dispositivos, como computadores, tablets e smartphones.
5. **Manutenibilidade** – O código deve seguir boas práticas de desenvolvimento, facilitando futuras manutenções e expansões.
6. **Disponibilidade** – O sistema deve estar disponível 99% do tempo, com manutenção programada apenas em horários de menor uso.

Casos de Uso (CU):

Caso de Uso 1: Cadastrar Usuário (Doador/Beneficiário)

- **Ator**: Usuário (Doador ou Beneficiário)
- **Descrição**: Um usuário acessa o sistema e se cadastra como doador ou beneficiário.

- **Fluxo Principal**:
    1. O usuário acessa a página de cadastro.
    2. O sistema solicita os dados pessoais (nome, e-mail, telefone, etc.).
    3. O usuário seleciona seu perfil (doador ou beneficiário).
    4. O sistema valida os dados e salva o cadastro.
    5. O usuário recebe uma confirmação de cadastro.
- **Fluxo Alternativo**:
    - Se houver erro de validação dos dados, o sistema informa o erro e solicita correção.

Caso de Uso 2: Registrar Doação

- **Ator**: Doador
- **Descrição**: Um doador cadastra uma nova doação no sistema.
- **Fluxo Principal**:
    1. O doador faz login no sistema.
    2. O doador acessa a área de "Registrar Doação".
    3. O sistema solicita informações sobre a doação (tipo de alimento, quantidade, data de validade, local de retirada).
    4. O doador preenche as informações e confirma o registro.
    5. O sistema salva a doação e notifica os beneficiários.
- **Fluxo Alternativo**:
    - Se o doador não preencher todos os campos obrigatórios, o sistema exibe uma mensagem de erro.

Caso de Uso 3: Consultar Doações Disponíveis

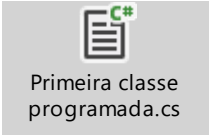
- **Ator**: Beneficiário
- **Descrição**: Um beneficiário consulta as doações disponíveis para retirada.
- **Fluxo Principal**:
    1. O beneficiário faz login no sistema.
    2. O beneficiário acessa a área de "Consultar Doações Disponíveis".

3. O beneficiário utiliza filtros (tipo de alimento, local, data) para refinar a busca.
4. O sistema exibe uma lista de doações disponíveis de acordo com os filtros aplicados.
5. O beneficiário escolhe uma doação e agenda a retirada.

- **Fluxo Alternativo**:
  - Se não houver doações disponíveis, o sistema exibe uma mensagem informando.

POO:

Primeira classe programada:



Uma classe principal implementada: