Inicialmente, foi desenvolvida a estrutura HTML responsável por exibir os campos de entrada de dados ao usuário. Esses campos incluem:Nome,E-mail,Mensagem,Campo opcional para inserção de imagem, que aceita tanto o arraste quanto a seleção direta de um arquivo.
Também foi aplicado estilo por meio de CSS para garantir uma interface amigável e responsiva. O HTML foi vinculado ao arquivo JavaScript index.js, que controla toda a lógica de interação do usuário com o sistema.

Validação e Consistência de Dados Foram implementadas validações para evitar o envio de dados duplicados. Sempre que o usuário insere um nome ou e-mail já cadastrado, uma mensagem de alerta é exibida logo abaixo do respectivo campo informando que os dados já estão registrados. Essa verificação acontece tanto no lado cliente quanto no servidor, garantindo dupla camada de proteção.

Interatividade com addEventListener A aplicação utiliza a função addEventListener para tornar os elementos da interface reativos. Eventos são associados a:Clique no botão de envio,Alterações nos campos de texto,Mudança de imagem no campo de upload
Esses eventos permitem que o sistema reaja dinamicamente às ações do usuário, por exemplo:Exibindo mensagens de erro ou sucesso,Atualizando o conteúdo de campos automaticamente,evitando envio com campos inválidos.

Atualização de Arquivo com atualizarNomeArquivo

A função atualizarNomeArquivo tem como objetivo atualizar dinamicamente o texto exibido no campo de upload da imagem. O comportamento implementado foi o seguinte: Antes da seleção, o campo mostrava um texto padrão: *"Arraste ou clique para inserir imagem"*.Após a escolha do arquivo, o texto é substituído pelo nome do arquivo.
O usuário pode clicar novamente para alterar a imagem.Por escolha de projeto, apenas uma imagem pode ser enviada por vez, e múltiplos uploads não são permitidos.

Envio de Dados com fetch e Feedback ao Usuário

Após a validação de todos os campos, o botão de envio dispara uma função que utiliza fetch() para enviar os dados ao servidor Python via método POST. Com base na resposta do servidor, uma mensagem de alerta é exibida no canto superior da tela:Se os dados forem enviados com sucesso, o usuário é informado.
Caso ocorra algum erro, como duplicidade de dados ou falha na comunicação, o sistema exibe uma mensagem apropriada.

Tratamento no Servidor (Python)

No back-end, o servidor é responsável por:Receber os dados via POST
Realizar um processo de GET interno para ler os dados já armazenados
Verificar se o nome ou e-mail já existem no arquivo de registro
Se já existirem, os dados não são salvos e o cliente é notificado
Se forem dados novos, eles são gravados na próxima linha disponível no arquivo
Quando há erro por dados duplicados, os campos são automaticamente limpos para que o usuário insira novas informações.
Foi implementado também um controle de contagem de caracteres no campo da mensagem. A quantidade máxima permitida é de 100 caracteres, sendo exibido ao usuário um contador dinâmico para que ele acompanhe em tempo real o espaço restante.

DESCRIÇÃO DE COMO RODAR OS CÓDIGOS

Todos arquivos devem estar na mesma pasta,após basta rodar :python .\servidor.py no terminal no caso usei o vscode e o servidor será iniciado.no meu caso estou usando um extenso do vscode chamada Live Server que após instalada é possível abrir a página web para testes.