Sobre o projeto

1. Link do Site

Link do site: https://radbios.com.br

2. Link do Repositório

Link do repositório: https://github.com/Radbios/contact-management.git

3. Principais problemas encontrados

Durante o desenvolvimento, alguns desafios foram enfrentados no que diz respeito ao design e à responsividade.

3.1. Desafios de Design e Responsividade

Problema: Uma grande dificuldade do projeto foi no desenvolvimento do design e na criação de uma interface responsiva. Isso exigiu bastante tempo para ajustar a aparência da aplicação.

Solução: Utilizei o MDB (Material Design for Bootstrap) para facilitar a criação de um design visualmente atraente e responsivo. Esse framework me ajudou a garantir que o layout se ajuste automaticamente em diferentes tamanhos de tela. Além disso, fiz ajustes manuais para melhorar a experiência do usuário, como o alinhamento de elementos e a adaptação de fontes e botões.

3.2. Gerenciamento de Features e Prazo Curto

Problema: Um grande desafio foi lidar com o prazo apertado para implementar todas as funcionalidades desejadas no projeto e, ao mesmo tempo, garantir a qualidade. Além dos requisitos obrigatórios, como autenticação, gerenciamento de contatos e exportação para CSV, eu acreditava que outras funcionalidades, como logs de ações, notificações e etc, eram essenciais para o bom funcionamento do sistema. A pressão do tempo dificultou o equilíbrio entre implementar essas features adicionais e garantir que o sistema estivesse funcional e com um código limpo e bem estruturado.

Solução: Para lidar com essa situação, foquei em implementar as funcionalidades mais críticas e em garantir a qualidade do código, ao invés de tentar adicionar muitas features no prazo curto. Acredito que ao manter o foco na qualidade das

features e na limpeza do código, consegui entregar um sistema funcional e eficiente, dentro do tempo disponível.

4. Deploy

A aplicação foi colocada no ar em uma VPS com CyberPanel, hospedada em um domínio adquirido pela Hostinger. Aqui estão os passos seguidos para colocar o sistema em produção:

4.1. Configuração do Servidor

1. **Criação do Website no CyberPanel**: Criei um novo website no painel de controle do CyberPanel, configurando o domínio para o projeto.

2. **Acesso via SSH**: Conectei à VPS via SSH para realizar o deploy do projeto.

4.2. Deploy do Projeto

1. **Clonagem do Repositório**: No servidor, acessei a pasta `public_html` e clonei o repositório e segui as etapas descritas no README do projeto.
2. **Configuração do .htaccess**: Criei um arquivo `.htaccess` no diretório raiz da aplicação para redirecionar todas as requisições para o `public/index.php` do projeto.

4.3. Finalização

Após esses passos, a aplicação foi colocada no ar com sucesso e está funcionando corretamente. Não foi necessário ajustar permissões de arquivos ou diretórios, pois o CyberPanel já vem com as configurações adequadas.

5. Escolhas no Desenvolvimento

5.1. Escolha do Projeto

Escolhi o Projeto 2 do teste em vez do Projeto 1 por diversas razões estratégicas. A principal delas foi a possibilidade de reutilizar o sistema no meu projeto pessoal de investimento. Como o Projeto 2 já possui a funcionalidade de gerenciamento de contatos, pude reaproveitar o que já estava implementado, economizando tempo e esforço no desenvolvimento do novo sistema.

5.2. Escolha do Laravel e da Estrutura MVC

Optei por utilizar o Laravel e a estrutura MVC devido à sua agilidade no desenvolvimento. Para um único desenvolvedor, como no meu caso, a estrutura MVC oferece uma organização clara do código, facilita a manutenção e a escalabilidade do sistema. Além disso, o Laravel fornece ferramentas integradas, como autenticação, validação e gerenciamento de banco de dados, que permitiram que eu focasse nas funcionalidades principais, garantindo um código limpo e de qualidade, sem precisar lidar com configurações complexas.