ESCOLA SENAI “HENRIQUE LUPO”

TECNICO EM ANALISE E DESENVOLVIMENTO DE SISTEMAS

ANA BEATRIZ CAMASSUTI FRANCISCATTO DA SILVA

BIANCA SIMONATO SCUPIN

GABRIEL CHAGAS FERNANDES DE MORAES

GUILHERME MITSUYUKI

JULIA DE BARROS RIBEIRO

EASY REQUEST

ARARAQUARA

2024

ANA BEATRIZ CAMASSUTI FRANCISCATTO DA SILVA

BIANCA SIMONATO SCUPIN

GABRIEL CHAGAS FERNANDES DE MORAES

GUILHERME MITSUYUKI

JULIA DE BARROS RIBEIRO

EASY-REQUEST

Trabalho de Conclusão de Curso/Projeto apresentado como requisito parcial para a obtenção do certificado de Desenvolvimento de Sistemas, da Escola SENAI “Henrique Lupo”.

Orientador(a): Prof. Alex Fernando Stocco,
Prof. Ivo Conceição Neto

ARARAQUARA

2024

Aos nossos pais,

Aos nossos professores,

E aos nossos amigos.

AGRADECIMENTOS

Agrademos de forma especial a Deus, por nos dar força e sabedoria durante essa jornada.

Aos nossos pais, que sempre nos apoiaram e acreditaram em nós, muito obrigado por estarem ao nosso lado.

Aos nossos professores, que nos ensinaram e nos ajudaram a crescer, somos gratos por todo o conhecimento compartilhado.

E aos nossos colegas de classe, que tornaram essa jornada muito mais divertida e significativa.

RESUMO

A manutenção escolar desempenha um papel fundamental na garantia do funcionamento adequado das instalações e equipamentos, com destaque para as manutenções preventiva, corretiva e preditiva. Contudo, na unidade escolar SENAI Araraquara, que é uma instituição de ensino técnico voltada para pessoas a partir de 14 anos (SENAI, 2024), adversidades como a organização entre funcionários diante os chamados de manutenção, que foram observados na unidade, têm o risco de comprometer a eficiência do setor.

Diante dessa problemática, este trabalho apresenta o desenvolvimento do sistema EASY REQUEST, uma plataforma digital voltada para otimizar a gestão das demandas de manutenção interna. A proposta visa facilitar o registro, acompanhamento e resolução de solicitações de serviço, oferecendo aos colaboradores acesso em tempo real ao status das demandas. Além disso, o sistema permitirá a categorização de serviços, designação de tarefas aos profissionais adequados e geração de relatórios detalhados.

O projeto busca proporcionar maior organização e eficiência nos serviços prediais, promovendo um ambiente mais seguro e colaborativo. O desenvolvimento utiliza Python, JavaScript e MySQL, com funcionalidades divididas em perfis de usuários (administrador, prestador de serviço e solicitante), integrando ferramentas de cadastro, gestão de chamados e relatórios.

Com base em entrevistas com supervisores de manutenção e feedbacks de usuários, o sistema será ajustado e aprimorado para atender às necessidades da instituição. Assim, o EASY REQUEST promete contribuir para a melhoria do ambiente escolar, fortalecendo a organização e a gestão de serviços de manutenção.

Palavras-chave: Manutenção; Organização; Sistema; Eficiência.

ABSTRACT

School maintenance plays a crucial role in ensuring the proper functioning of facilities and equipment, with an emphasis on preventive, corrective, and predictive maintenance. However, communication and organizational issues among staff, as observed at the SENAI Araraquara unit, hinder the efficiency of the department.

In light of this issue, this work presents the development of the EASY REQUEST system, a digital platform designed to optimize the management of internal maintenance requests. The goal is to streamline the registration, tracking, and resolution of service requests, providing employees with real-time access to the status of these demands. Additionally, the system will enable service categorization, task assignment to appropriate professionals, and the generation of detailed reports.

The project aims to bring more organization and efficiency to building services, fostering a safer and more collaborative environment. The development uses Python, JavaScript, and MySQL, with functionalities divided into user profiles (administrator, service provider, and requester), integrating tools for registration, call management, and reporting.

Based on interviews with maintenance supervisors and user feedback, the system will be adjusted and improved to meet the institution’s needs. Thus, EASY REQUEST promises to contribute to the improvement of the school environment by strengthening the organization and management of maintenance services.

Keywords: Maintenance; Organization; System; Efficiency.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

OS	Ordem de Serviço
SHA-256	Secure Hash Algorithm 256 bits
AJAX	Asynchronous JavaScript and XML
SQL	Structured Query Language
WEB	World Wide Web
MySQL	Structured Query Language
CPF	Cadastro de Pessoa Física
IOS	Operational System
SAP	System Analysis Program Development
MacOS	Macintosh Operating System

SUMÁRIO

1 INTRODUÇÃO

O setor de manutenção em uma escola é o departamento responsável por ser o agente que atua diretamente a funcionalidade de diferentes áreas de uma escola para garantir que todas as instalações e equipamentos funcionem corretamente, em especifico um cenário de ambiente escolar. Esse setor realiza três tipos principais de manutenção: a preventiva, a corretiva, e a preditiva.

A manutenção preventiva consiste em um conjunto de ações que têm objetivo de evitar falhas e garantir o funcionamento contínuo dos equipamentos e instalações. Em um ambiente escolar, inclui a realização de inspeções em laboratórios, salas de aula e áreas comuns, como a verificação de equipamentos didáticos, sistemas elétricos e hidráulicos. (https://www.produttivo.com.br/blog/tipos-de-manutencao-quais-suas-diferencas/)

A manutenção corretiva atua quando surgem falhas inesperadas nos equipamentos ou na infraestrutura. Exemplos em uma escola técnica incluem a reparação de máquinas que apresentaram defeitos, consertos em sistemas de climatização, ou reparos em estruturas como pisos e paredes danificadas. (https://www.produttivo.com.br/blog/tipos-de-manutencao-quais-suas-diferencas/)

A manutenção preditiva envolve o monitoramento constante das condições dos equipamentos e das instalações, utilizando tecnologias e métodos analíticos para prever falhas antes que elas ocorram. Em uma unidade escolar técnica, pode incluir técnicas como a medição de vibrações em máquinas, o uso de termografia para identificar pontos quentes em sistemas elétricos. (https://www.produttivo.com.br/blog/tipos-de-manutencao-quais-suas-diferencas/)

Recentemente, tem ficado evidente a divergência no cenário dos setores de manutenção e comunicação que ocorre internamente entre funcionários nas instituições de ensino, em específico visando a instituição escola SENAI Araraquara. É nítido que existe uma carência de uma comunicabilidade mais precisa, já que as áreas de manutenção devem ter uma maior relevância, pois nesse sentido, haverá uma melhor convivência na vida profissional de todos os indivíduos presentes na instituição colocada em tese e também tornará o ambiente mais seguro de acidentes e incidentes que podem causar danos aos profissionais e/ou envolvidos.

Por certo, fica claro afirmar que uma vez que sucede um problema/incidente no local de trabalho e estudo, a necessidade de um imediato meio de solucionar tal evento, torna-se indispensável para assim decorrer um ambiente mais organizado e seguro e que, consequentemente, um local com menos divergências entre funcionários pela escassez de comunicação e na redução de adversidades que não são atendidas, que são esquecidas ou que têm uma delonga para serem realizadas.

Com isso, surgiu a pergunta problema do presente trabalho:

Como otimizar a organização das altas demandas nos serviços de manutenção do SENAI?

Portanto, diante do cenário de estudo sobre a demanda necessária e da ideia principal de projeto, o presente trabalho consiste em um sistema destinado a otimizar a organização e a gestão das solicitações de serviços de manutenção interna dentro da instituição citada, e por sua vez, os encaminhamentos de serviço aos funcionários prestadores de serviço da manutenção. O objetivo é uma ferramenta para facilitar a comunicação entre os funcionários de manutenção e os demais colaboradores da unidade escolar que necessitam de um serviço a ser efetuado pelo mesmo setor. O sistema seria um local eficiente onde os

colaboradores irão reportar e acompanhar a resolução dos problemas prediais e ter acesso às informações em tempo real do andamento da realização do serviço. Para isso, o programa permitirá o cadastro dos colaboradores de acordo com suas funções e especializações dentro da instituição, facilitando assim, a identificação do profissional mais adequado para cada tarefa a ser efetuada.

A plataforma presente não visa somente resolver os problemas de maneira mais ágil, mas também melhorar a organização e a gestão dos serviços prediais, contribuindo para um ambiente de trabalho mais fluido e seguro.

Objetivo Geral

O principal objetivo do projeto consiste em desenvolver uma plataforma para otimizar a organização e gestão de manutenção interna em uma unidade escolar. O sistema busca integrar e aprimorar a troca de informações entre os colaboradores envolvidos no processo da realização da manutenção, garantindo uma organização eficiente das solicitações e um acompanhamento dos serviços prestados, elevar a satisfação dos funcionários ao oferecer uma ferramenta intuitiva e eficaz para o gerenciamento das solicitações.

Objetivos Específicos

1. Realizar entrevista com o supervisor de manutenção para entender o funcionamento da comunicação entre os funcionários e a organização do setor de manutenção.

2. Levantar a ideia do sistema, abordando como ele auxiliará na manutenção predial e gestão de serviços, utilizando Python, JavaScript e banco de dados MySQL.

3. Desenvolver a lógica do sistema e iniciar o protótipo inicial, incluindo esboço e wireframe da plataforma.

4. Iniciar a programação com funcionalidades básicas como cadastro de funcionários (Administrador, Prestador de Serviço/Solicitante e Apenas Solicitante), e gestão de solicitações, categorização de serviços e chamados.

5. Aperfeiçoar o sistema para permitir o detalhamento das solicitações, incluindo tipo de serviço, local e urgência, além da gestão de solicitações pelo supervisor.

6. Incluir funcionalidades para criação de relatórios detalhados e histórico de serviços, facilitando o controle das atividades realizadas.

7. Realizar testes de usabilidade com usuários, coletando feedback para melhorar a experiência, dividindo os resultados por categorias.

8. Implementar a versão final do sistema com ajustes baseados nos testes, incluindo painel administrativo e ferramentas de controle e geração de relatórios.

1.1 JUSTIFICATIVA

Em suma, nós como alunos da instituição SENAI, com a conscientização de que há uma possibilidade de um melhor atendimento as adversidades que surgem no dia-a-dia dos funcionários gerais com uma maior ênfase ao setor da manutenção e no processo de realização de serviços internos reconhecemos que o sistema é necessário e tem relevância no cenário atual. Por fim, o sistema EASY REQUEST tem potencial para suprir as necessidades que estão em evidência no processo de efetuação de serviço e organização, um ambiente mais organizado e inclusivo socialmente ao se dar maior visibilidade ao setor que foi abordado.

2 PLANEJAMENTO DE PROJETO

2.1 Entrevista

No decorrer do desenvolvimento do nosso sistema de apoio à manutenção, foi realizada uma entrevista com Cláudio, Supervisor de Manutenção do SENAI de Araraquara. A seguir, apresentamos as principais informações extraídas dessa entrevista, que contribuíram para o entendimento dos processos atuais de manutenção e das necessidades do setor.

1. Descrição do Trabalho e Fluxo de Atividades

Cláudio é responsável pela gestão e execução dos serviços de manutenção no SENAI de Araraquara, supervisionando uma equipe composta por quatro funcionários, com formações em áreas distintas: Hidráulica, Elétrica, Pintura e Alvenaria. A principal função do supervisor é garantir que os problemas de manutenção sejam resolvidos de forma eficiente e que as tarefas sejam distribuídas corretamente entre os membros da equipe. Ele descreveu que, inicialmente, as solicitações de serviços chegam até ele por e-mail e WhatsApp, sendo que os pedidos não possuem uma classificação clara de prioridade ou urgência, o que dificulta a organização das tarefas.

2. Organização do Trabalho

No SENAI de Sertãozinho, onde Cláudio trabalhou por sete anos, ele utilizava o Planner da Microsoft Office para gerenciar as demandas de manutenção. A utilização dessa ferramenta ajudava na organização das atividades, mas, de acordo com ele, não resolvia completamente os desafios relacionados à clareza na priorização e na distribuição de tarefas. No SENAI de Araraquara, a estrutura

organizacional conta com cinco OPPs (Organizações de Processos de Produção), cada um responsável por um setor específico, o que influencia diretamente na solicitação de serviços. No entanto, a falta de uma comunicação clara sobre a urgência de cada serviço dificulta o gerenciamento das demandas.

3. Desafios na Distribuição de Tarefas

Cláudio destacou que a distribuição de tarefas entre os membros da equipe seria mais eficiente se houvesse uma ferramenta que permitisse classificar e priorizar as demandas de forma mais automatizada. Além disso, a comunicação sobre a urgência e os detalhes dos serviços a serem realizados precisa ser mais clara para evitar um retrabalho.

4. Uso de Ferramentas e Plataformas de Apoio

Atualmente, Cláudio não utiliza uma plataforma específica para gerenciar as tarefas da equipe, o que torna o processo de acompanhamento mais complexo. Embora o uso de e-mail e WhatsApp seja comum, ele reconhece que uma plataforma mais robusta poderia facilitar a comunicação, a organização e o controle das atividades de manutenção.

5. Considerações finais da entrevista com supervisor de manutenção

A principal necessidade identificada foi a criação de uma plataforma que centralize as solicitações, classifique as prioridades e melhore a comunicação entre os membros da equipe. Além disso, o gerenciamento das tarefas seria significativamente facilitado com a automação do processo de distribuição e o monitoramento das demandas em tempo real. Essas informações foram essenciais para moldar as principais ideias do sistema, garantindo que ele atenda às necessidades de Cláudio e de outros profissionais da área de manutenção.

2.2 Levantamento de Requisitos

2.2.1 Público-alvo

O projeto Easy Request tem como principal público os profissionais da área de manutenção da escola SENAI Araraquara.

O projeto consiste na presença de apenas três indivíduos responsáveis pela sua execução, sendo eles os indivíduos: supervisor de manutenção, técnico de manutenção e solicitante.

- Supervisor de manutenção**:** Refere-se ao profissional supervisor responsável pela manutenção geral do SENAI.
- Técnico de manutenção**:** Refere-se ao profissional responsável pela realização dos serviços de manutenção do SENAI.
- Solicitante**:** Refere-se a qualquer profissional o qual requerer de um chamado de efetuação de serviço na manutenção interna do SENAI.

2.2.2 Requisitos Funcionais

Os requisitos funcionais são todos os problemas e necessidades que devem ser atendidos e resolvidos pelo software por meio de funções ou serviços. Tudo o que for relacionado a uma ação a ser feita é considerado uma função. Também é importante lembrar que quanto menos ambíguos e mais objetivos forem os requisitos funcionais, maior será a qualidade do software gerado pelo time de qa (quality assurance).

2.2.3 Requisitos Não-Funcionais

Os requisitos não funcionais são todos aqueles relacionados à forma como o software ou aplicativo web tornará realidade os que está sendo planejado. Ou seja, enquanto os requisitos funcionais estão focados no que será feito, os não funcionais descrevem como serão feitos.

2.2.4 Regras de Negócio

Uma regra de negócio descreve um aspecto do negócio, definindo ou restringindo tanto sua estrutura quanto seu comportamento. Ou ainda: Uma regra define uma diretriz para cada contexto específico de um negócio, sobre qual deve ser o resultado esperado para cada ação ou decisão.

PRIORIDADE DOS REQUISITOS

Para estabelecer a prioridade dos requisitos, foram adotadas as denominações: essencial, importante e desejável.

- Essencial: Tudo aquilo que é fundamental para o funcionamento do projeto. Sem as prioridades essenciais, o sistema não funcionaria.
- Importante: Tudo aquilo que seria importante que fosse inserido no projeto, porém o sistema pode funcionar normalmente sem as prioridades importantes.
- Desejável: Tudo aquilo que seria apenas um incremento para o aperfeiçoamento do sistema, mas não interfere no objetivo principal do projeto.

3 FERRAMENTAS E LINGUAGENS DE PROGRAMAÇÃO

Após a finalização do protótipo, a efetiva implementação e inicio do sistema foi iniciada.

Portanto, o presente trabalho consiste em uma busca de auxiliar e tornar mais eficiente o sistema atual, apresentando uma comunicação mais eficaz e mais rápida entre os funcionários da seguinte unidade escolar. Ademais, também será apresentado um histórico fixo de cada chamado que auxiliará o supervisor a inserir informações na ordem de serviço ou consulta-los após um longo período com fácil acesso, caso houver essa necessidade.

O presente sistema foi feito a partir da utilização da linguagem de marcação HTML, a linguagem de estilização CSS, as linguagens de programação JavaScript e Python e o banco de dados MySQL. A organização para efetuação do projeto foi feita a partir de realização de cada um dos requisitos funcionais e não-funcionais, com cronograma feito a partir de cada uma das partes do projeto – divididas em front-end, back-end, documentação, banco de dados, e wireframe na plataforma Notion que ia sendo atualizado – com iniciado, em andamento ou concluído – conforme o projeto era realizado.

3.1 Microsoft Azure

Microsoft Azure é uma plataforma em nuvem que oferece serviços de hospedagem, armazenamento e outras ferramentas online. No projeto, o Azure é

utilizado para hospedar o sistema, possibilitando que ele esteja disponível na internet para os usuários de maneira segura e confiável.

3.2 Visual Studio Code

Visual Studio Code é um editor de código com vários recursos que facilitam o desenvolvimento. Utilizamos o Visual Studio Code como ambiente principal para escrever, organizar e testar todo o código do sistema de forma prática e eficiente.

3.3 Notion

Notion é uma ferramenta de organização que facilita o gerenciamento de tarefas e projetos. Utilizamos o Notion para documentar o progresso do projeto,

3.4 Github

GitHub é uma plataforma que hospeda códigos de programação e usa o sistema de controle de versão Git. Ele permite que o projeto seja armazenado e integrado de forma que todos os integrantes do projeto tenham acesso ao mesmo projeto e ao mesmo tempo, ajudando a acompanhar as mudanças feitas no código garantindo que todos os desenvolvedores tenham uma melhor organização e segurança.

3.4.1 Git

O Git é uma ferramenta utilizada para controlar as mudanças em arquivos, especialmente em projetos de programação. Ele permite que várias pessoas trabalhem ao mesmo tempo, permitindo que todos os integrantes de um projeto tenham acesso as alterações foitas no código. Com o Git é possível ver o histórico de alterações, restaurar versões antigas, criar ramificações (branches) para

desenvolver novos recursos sem interferir no código principal. É uma ferramenta essencial para o versionamento de código em desenvolvimento colaborativo.

3.5 FRONT-END

O front-end é a camada visível do sistema, responsável pela interação direta do usuário com a aplicação. Ele engloba tudo o que é exibido na tela, como layout, design e elementos interativos. Utiliza tecnologias como HTML para estruturar o conteúdo, CSS para estilização e JavaScript para tornar as páginas dinâmicas e responsivas. O principal objetivo do front-end é criar interfaces intuitivas e agradáveis, garantindo uma experiência fluida e eficiente para o usuário.

3.5.1 HTML

HTML é a linguagem de marcação que organiza a estrutura dos elementos do sistema, como cabeçalhos, parágrafos e imagens. No projeto, o HTML cria a “base”

3.5.2 CSS

CSS é a linguagem que define a aparência visual do site, ajustando cores, tamanhos de fonte e espaçamento dos elementos. Usamos CSS para tornar o site mais bonito e organizado, garantindo uma identidade visual uniforme.

3.5.3 JavaScript

O JavaScript é uma linguagem que permite adicionar elementos interativos ao site. No sistema do Easy Request, ele é usado para tornar a navegação mais

dinâmica e agradável, respondendo às ações do usuário, como cliques em botões ou mudanças na página.

3.5.4 Figma

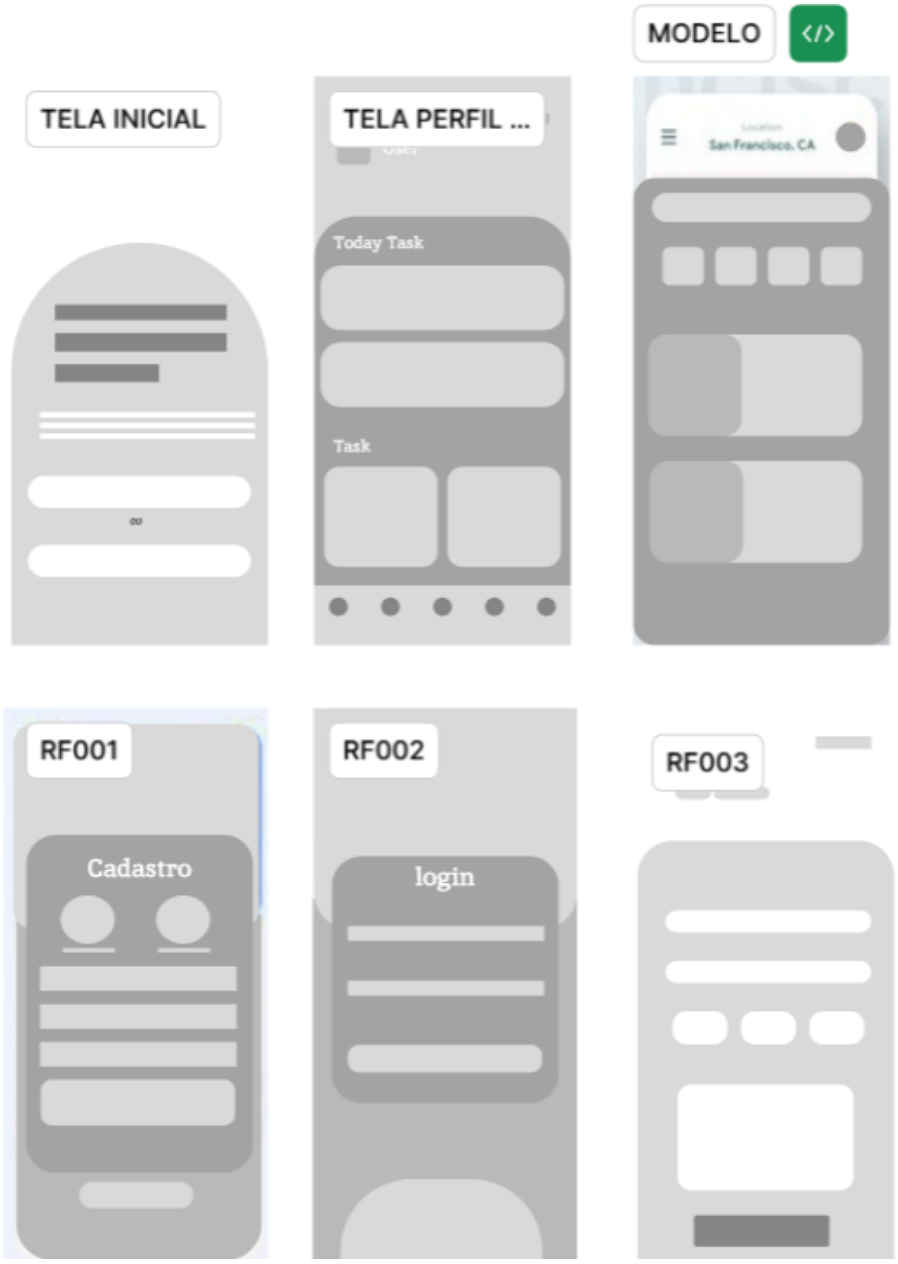
Figma é um software de design que permite criar layouts de sites de forma colaborativa. No projeto, ele foi usado para planejar a aparência de cada página do site, ajudando na definição de cores, tamanhos e posicionamento de cada elemento visual através do wireframe e protótipo que foi realizado no começo do planejamento de layout, que é a organização inicial dos elementos do projeto.

3.5.4.1 Wireframe

O wireframe foi uma ferramenta usada para que fosse feito em forma de rascunho e esboço as primeiras ideias da interface do sistema, utilizando a plataforma Figma e os elementos presente nela.

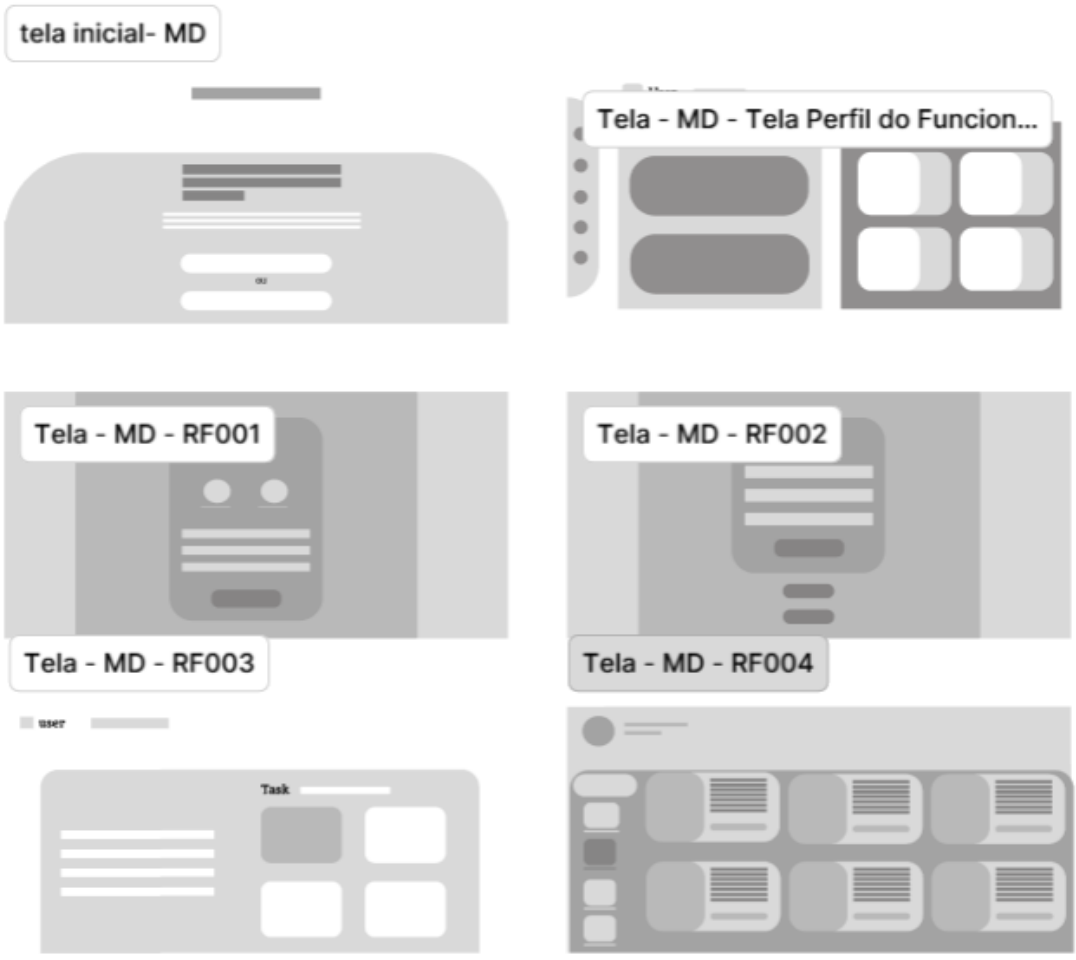
Assim, foi feito o esboço da interface de cada uma das telas de acordo com os requisitos funcionais – definidos anteriormente ao início do planejamento – em telas de tamanho desktop – normalmente utilizadas em computadores e notbooks – e também em telas de tamanho mobile – utilizadas em telas de celulares.

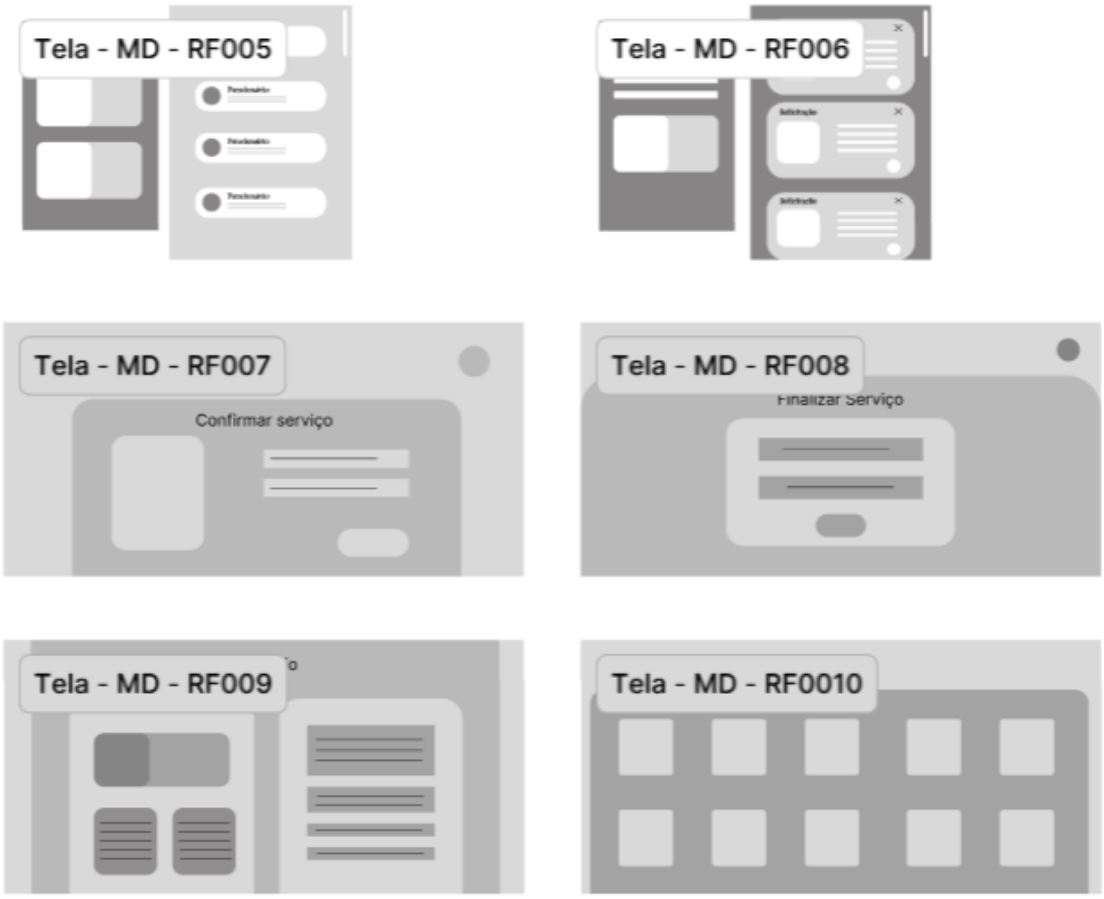
Abaixo refere-se à versão mobile do wireframe.


MODELO
TELA INICIAL
TELA PERFIL ...
Today Task
Task
RF001
Cadastro
RF002
login
RF003


RF004
Funcionário
Funcionário
Funcionário
Funcionário
RF006
Solicitação
RF007
Confirmar finalização de serviço
RF008
RF009
RELATÓRIO
RF010
Histórico de Tarefas

Abaixo refere-se à versão de tamanho desktop do wireframe.

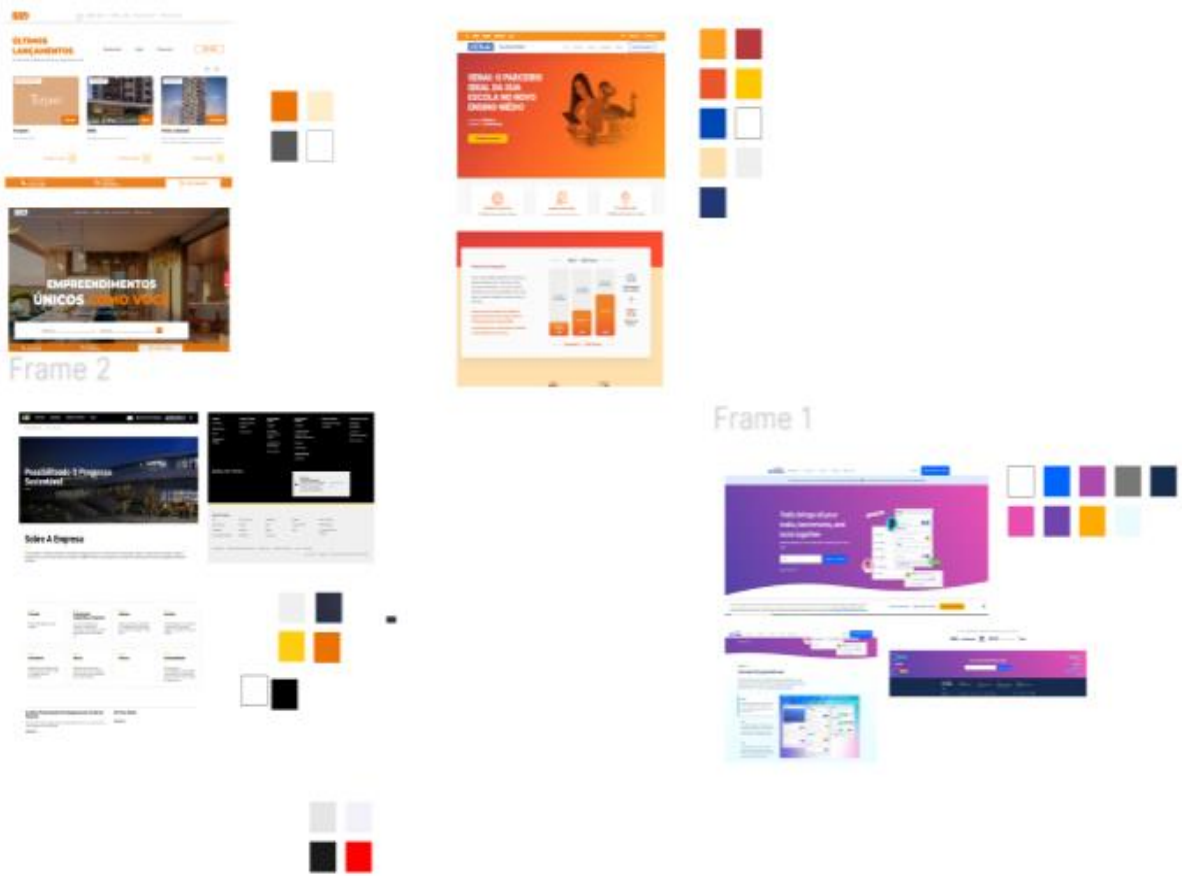




3.5.4.2 Estudo de cores

A seleção de cores que deveriam ser utilizadas no sistema foi inspirada nas ferramentas e elementos que se relacionam com os técnicos da manutenção - capacete, fita métrica, placas de aviso e outros - sendo elas laranja, azul marinho e amarelo. Laranja e amarelo são cores análogas, ou seja, no círculo cromático elas estão em sequência. Para dar um contraste entre essas cores, foi decidido a cor complementar ao laranja, o azul marinho, que se encontra do lado oposto ao laranja no círculo cromático. As demais cores utilizadas como cinza, vermelho e verde são

auxiliadoras na paleta de cores do sistema e visam a harmonia e melhor experiência do usuário.



3.5.4.3 Protótipo

O protótipo se refere à versão colorida do wireframe, versão aprimorada cujo também é a versão de esboço que mais se aproxima da que foi implementada no sistema – realizado também na plataforma Figma e em tamanhos desktop e mobile.

Abaixo refere-se a versão de tamanho mobile do protótipo.


TELA INI...
RF001
RF002
BEM-VINDO AO EASY REQUEST
Gerencie seus chamados de forma simples e rápida
Você já tem uma conta?
Entrar
ou
Cadastre-se
Cadastro
Login
RF003
RF004
RF004A-d...
Área de Solicitação
Predio A , sala 31
Encaminhar
RF006
RF006 - TEL...


RF005
Pintor
Disponível
Encanador
Disponível
Eletricista
Em Serviço
Alvenéiro
Disponível
Prioridade:
Baixa Urgencia
Média Urgencia
Urgencia
Boa Tarde!
Josefino Alves
Serviços de Aberto
13h45
Bloco B | Sala 22
15h02
Bloco A | Sala 10
Serviços Encaminhados
QUA 09
Bloco B | Sala 09
QUI 10
Bloco A | Sala 07
Quer fazer uma solicitação?
Solicite com o botão abaixo!
Fazer Solicitação
Boa Tarde!
Úliver Santana
Serviços em Aberto
13h45
Bloco B | Sala 22
15h02
Bloco A | Sala 10
Histórico de Solicitações
12h46 - Sáb 06 Set
Bloco C | Sala 21
Quer fazer uma solicitação?
Solicite com o botão abaixo!
Fazer Solicitação
Boa Tarde!
Josefino Alves
Serviços em Aberto
RF007
Administrador
Confirmar finalização de serviço
Escolha uma das opções:
Não foi possível realizar a tarefa
Foi realizado mas necessita de outros cuidados
Tarefa foi totalmente realizada.
Inserir informações adicionais (caso houver)
Digite aqui
Finalizar tarefa
RF008

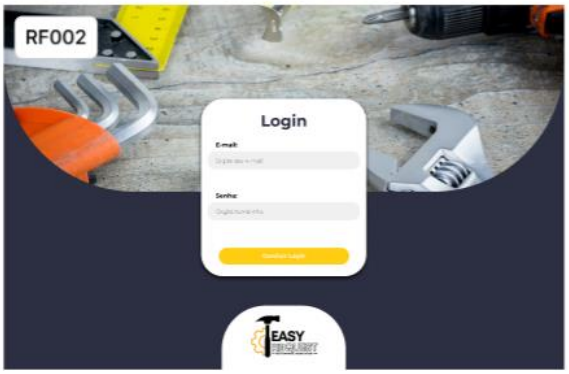

Voltar
status
Tipo
Aonde esta
Descrição
Encerrar chamado
Encerrar chamado
RF009
Administrador
Relatório de serviço
RF010
órico de serviços
Heitor Lima
Josefino Alves
Rafael Augusto
Ver mais
Ver mais
Ver mais

Abaixo refere-se a versão de tamanho desktop do protótipo.

Tela Inicial

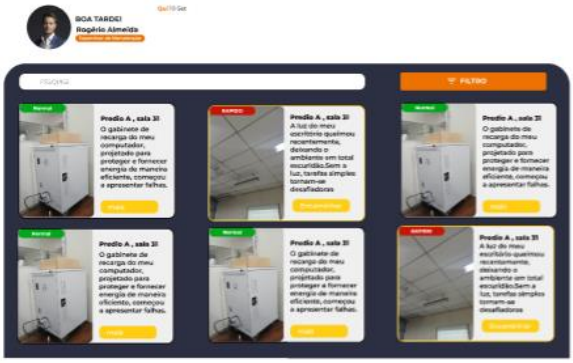
RF001

RF002



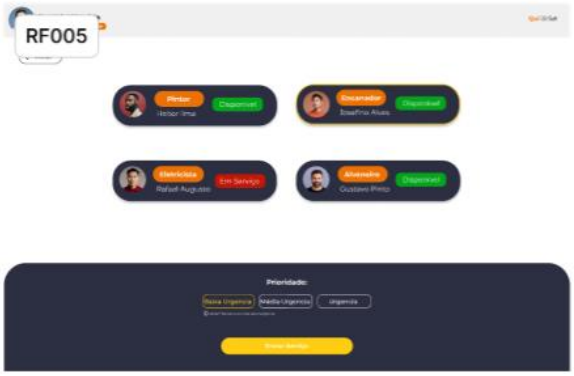
RF003

RF004



RF004A-detlSolic

RF005



RF006

3.5.5 Font Awesome

Font Awesome é uma biblioteca de ícones que podem ser usados em sites. No projeto, esses ícones melhoram a navegação e dão um toque visual mais interessante, além de ajudar a indicar certas ações para os usuários, como “voltar” ou “salvar”.

3.5.6 Google fonts

Google Fonts é uma coleção de fontes gratuitas disponibilizada pelo Google. No projeto, usamos o Google Fonts para escolher fontes que combinam com a identidade visual do site, proporcionando uma leitura confortável e mantendo o estilo em diferentes dispositivos e navegadores.

3.5.7 Scrollreveal

O ScrollReveal faz parte do front-end do sistema. Ele é uma biblioteca de JavaScript que ajuda a criar animações quando elementos entram na área visível da tela (o viewport) enquanto o usuário rola a página. Ele é usado para adicionar interatividade, destacando conteúdos e elementos da interface.

3.5.8 Sweet Alert

O Sweet Alert é uma biblioteca responsável pela criação de alertas e notificações estilizadas, substituindo os alertas padrão do navegador. Esta biblioteca tem um estilo moderno personalizável, fácil de usar e compátível com todos os navegadores atuais.

3.6 BACK-END

Back-end refere-se à parte do sistema que lida com o processamento dos dados, a lógica de negócios e a comunicação com o banco de dados. É o "motor" que faz o sistema funcionar, sem o qual a interface do front-end não teria dados para exibir ou processar. O back-end é implementado com linguagens de programação como Python, Java, PHP, Node.js, entre outras, e usa frameworks e bibliotecas específicas para gerenciar servidores, APIs e bases de dados. A principal

responsabilidade do back-end é garantir que a aplicação seja escalável, segura e eficiente.

3.6.1 Python

O Python é uma linguagem de programação usada no desenvolvimento de sistemas e sites por ser fácil de entender e trabalhar. No presente projeto, o Python é responsável pelo funcionamento interno do site, conectando a lógica do sistema ao banco de dados e às outras funcionalidades.

da plataforma, definindo o que aparece na tela e como as informações são apresentadas ao usuário.

3.6.1.1 Flask

Flask é um micro framework em Python usado para criar aplicativos web de forma simples e rápida. Ele fornece as ferramentas básicas para desenvolver sites e APIs, permitindo que você adicione apenas o que precisar. Por exemplo, com o Flask, você pode definir rotas para responder a diferentes URLs e criar páginas dinâmicas ou sistemas de back-end facilmente.

3.6.1.2 FIREBASE

O Firebase é uma plataforma desenvolvida pelo Google para diversas ferramentas e serviços para o desenvolvimento de aplicativos móveis e web. Ele torna a criação, gerenciamento e expansão de aplicativos mais fácil e rápida, oferecendo soluções prontas para back-end, como notificações push.

3.6.1.2.1 FIREBASE CLOUD MESSAGING(FCM)

O Firebase Cloud Messaging é um serviço oferecido pelo Firebase que possibilita o envio de notificações para dispositivos Android, iOS e navegadores web. Com ele é possível realizar o envio de notificações personalizadas ou mensagens de dados diretamente para o aplicativo/aparelho do usuário.

Esse serviço também permite o envio de notificações de duas formas diferentes: em primeiro plano e em segundo plano. Com o envio em primeiro plano é possível personalizar o pop-up de notificações de diversas formas diferentes, criando assim uma melhor interação com o usuário. Já o envio em segundo plano permite que o usuário receba notificações mesmo com o aplicativo fechado, tornando o envio de notificações mais eficiente.

3.6.2 Banco de Dados

Um banco de dados é uma ferramenta usada para armazenar, organizar e gerenciar informações de forma estruturada. Ele permite que dados sejam facilmente acessados, atualizados e manipulados por sistemas de software. Em um site, o banco de dados armazena as informações que os usuários interagem, como registros de chamados, usuários, histórico de manutenção, status das tarefas, entre outros dados relevantes.

No contexto do presente projeto, ele atua no cadastro de cada funcionário, guardando as informações fornecidas por cada funcionário. Ele guarda também todas as informações necessárias da unidade escolar, como os blocos e salas.

O banco de dados atua também quando um chamado de manutenção é registrado no sistema com as informações como a descrição do problema, a data de abertura, a prioridade e o responsável pela tarefa. À medida que os funcionários da

manutenção atualizam o status do chamado, como "em andamento" ou "resolvido", essas informações são modificadas no banco de dados em tempo real.

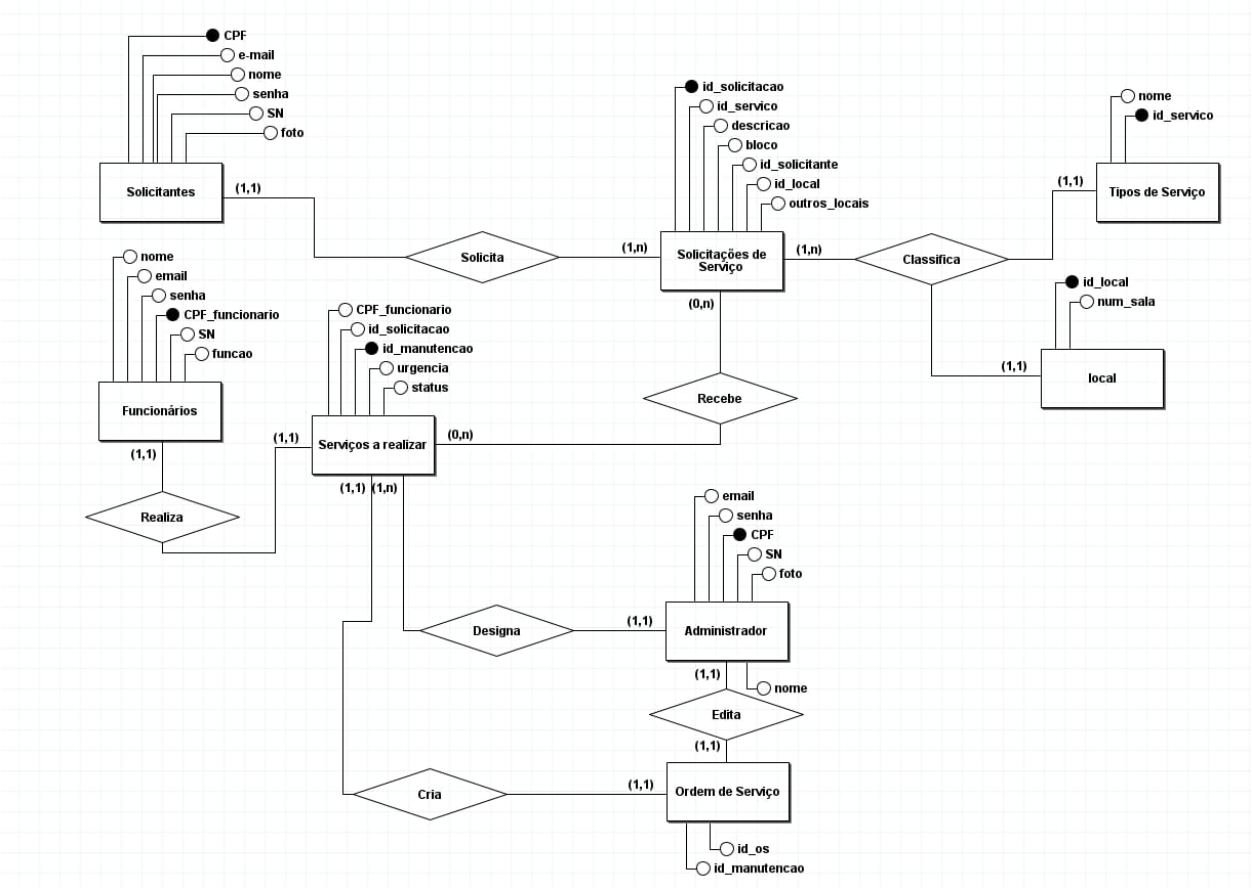
Além disso, o banco de dados permite que os o supervisor de manutenção acesse rapidamente o histórico de manutenção, garantindo que mesmo depois de encerrados, as informações sobre cada chamando não serão perdidas. Ele também é responsável por gerar os relatórios ao final de cada chamado para auxiliar o supervisor de manutenção na criação da ordem de serviço.

3.6.3 Diagrama de caso de uso

No projeto, o Diagrama de Casos de Uso é uma representação gráfica das funcionalidades principais de um sistema e dos atores que interagem com ele. Ele organiza e ilustra de forma clara as ações que cada tipo de usuário realiza no sistema, possibilitando uma visão ampla das interações e funcionalidades esperadas, o que é essencial para a compreensão das operações do sistema.

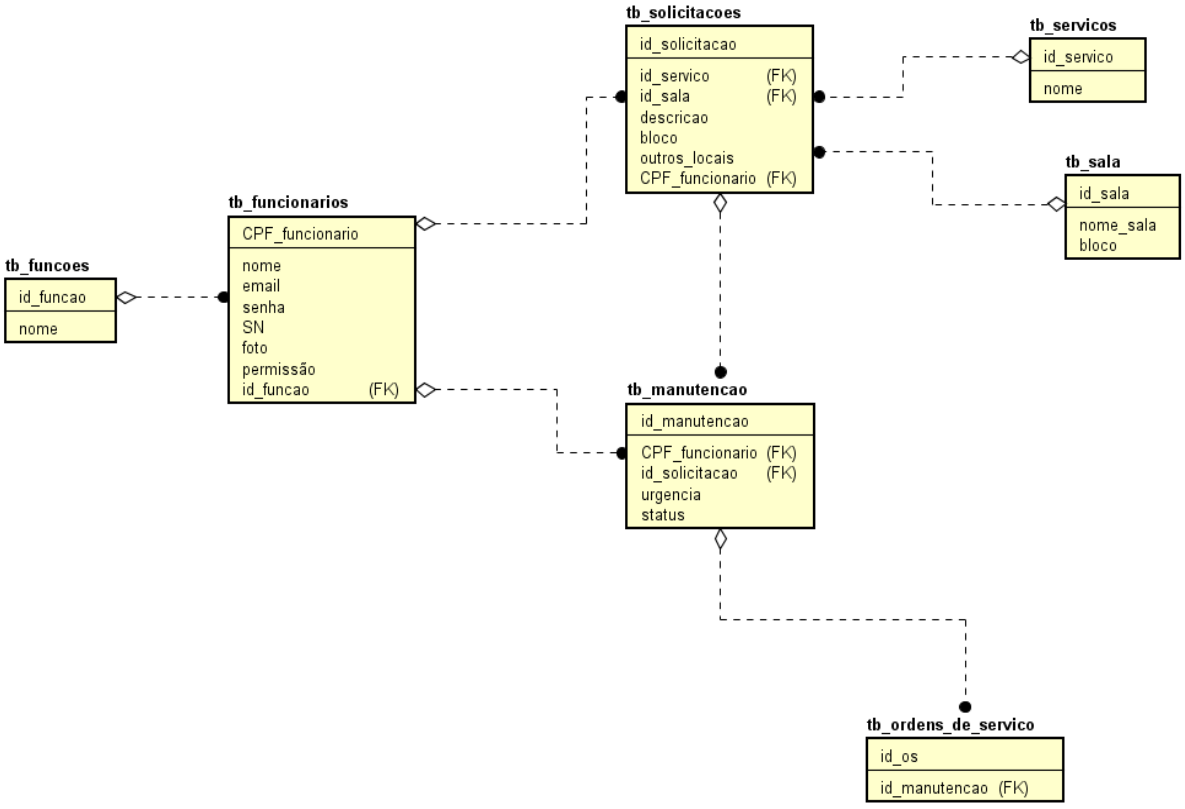
3.6.3.1 MER (Modelo Entidade Relacionamento)

No presente projeto o Modelo Entidade-Relacionamento (MER) organiza e define, de maneira teórica, as principais informações do sistema, identificando entidades, atributos e relações entre elas. Esse modelo permite descrever quais dados são relevantes e como esses dados se interligam, sendo essencial para o planejamento de bancos de dados estruturados e consistentes.



3.6.3.2 DER (Diagrama Entidade-Relacionamento)

O Diagrama de Entidade-Relacionamento (DER) é uma representação visual do MER, utilizado para ilustrar a estrutura do banco de dados. No DER, as entidades, seus atributos e os relacionamentos entre elas são organizados de forma gráfica por meio de retângulos, elipses e losangos, proporcionando uma visão clara e objetiva da organização e da conexão entre dados no banco planejado.



3.6.4 MySQL WorkBench

MySQL Workbench é uma ferramenta visual para criar e gerenciar o banco de dados. Neste projeto, ele é usado para armazenar dados importantes, como informações dos usuários e conteúdo de dados internos do sistema, facilitando o acesso e a organização das informações.

4 SISTEMA EASY REQUEST

4.1 FUNCIONALIDADES IMPLEMENTADAS

- Somente o supervisor da manutenção deve passar as ordens de serviço que foram solicitadas para os prestadores de serviços qualificados na área solicitada. Essa regra visa garantir a organização dos prestadores de serviço, para que não existam problemas com trocas de funções nas áreas que não são intituladas aos mesmos.
- O técnico de manutenção só pode aceitar uma tarefa por vez, e só voltar a aceitar outra depois de finalizar a anterior. A regra visa garantir que o técnico de manutenção não se comprometa a executar vários serviços de uma vez, deixando de cumprir todas de forma excepcional.
- Quando o técnico de manutenção iniciar um serviço, os outros serviços devem ficar em um local diferente na tela principal dele, mostrando o serviço que o funcionário está fazendo e os outros que ele precisa fazer. Se o técnico de manutenção tentar aceitar outro serviço, será exibida uma mensagem de impedimento.
- Todos os usuários incluindo o administrador terão acesso às ordens de serviço e saberão como está o status do serviço. A regra visa garantir que os usuários de cada bloco e o supervisor da manutenção vejam todas as ordens de serviços e vejam os status de cada serviço como: À Fazer, Fazendo e Feito. Esta função auxilia na organização das demandas para com todos os usuários de cada bloco e o supervisor da manutenção.

- No momento da designação de serviço será gerado um relatório que irá facilitar ao supervisor de manutenção a preencher uma Ordem de Serviço que será enviada para a sede do SENAI. Esta função visa seguir as ordens da sede da empresa, oferecendo um relatório do serviço executado.
- Quando o supervisor da manutenção finalizar o serviço de um técnico de manutenção, será criado o relatório automaticamente de acordo com as informações do serviço.

4.1.1 Cadastro de Usuário

A função de cadastrar usuário é um processo fundamental para o funcionamento do programa. O cadastro pode ser realizado por três tipos de contas: administrador (supervisor de manutenção), prestador de serviço (técnico de manutenção) e usuário solicitante (qualquer funcionário que precise fazer solicitações). Essa etapa é importante para que todos possam criar suas contas e utilizar o sistema.

Para realizar o cadastro, o usuário deve fornecer algumas informações pessoais, incluindo: nome completo, CPF, e-mail, senha e área de atuação.

Se o cadastro não for concluído corretamente, o sistema deve exibir uma mensagem informando que houve uma falha, e o usuário será direcionado de volta à página de cadastro para tentar novamente. Por outro lado, se o cadastro for bem-sucedido, uma mensagem confirmará que o registro foi realizado com sucesso.

Após a conclusão do cadastro, o sistema redirecionará o usuário para a página inicial de sua conta, onde ele terá acesso a todas as funcionalidades disponíveis no programa.

4.1.2 Login

A função de realizar login é um procedimento fundamental para o funcionamento do programa. Todos os usuários devem passar por esse processo para acessar suas contas e utilizar as funcionalidades disponíveis de acordo com seu nível de acesso.

Para efetuar o login, o usuário precisa ter completado o cadastro previamente, garantindo que suas informações pessoais estejam registradas no sistema. Durante o login, o usuário deve fornecer dois dados essenciais: seu e-mail e sua senha.

Se o login não for bem-sucedido, o sistema deve exibir uma mensagem informando que o e-mail ou a senha estão incorretos, permitindo que o usuário volte à página de login para tentar novamente. Por outro lado, se o login for realizado corretamente, o sistema deve mostrar uma mensagem confirmando que o processo foi bem-sucedido.

Uma vez que o login é concluído sem erros, o usuário é direcionado para a página principal do sistema, que é adaptada conforme seu perfil de acesso. Assim, ele pode navegar na plataforma e aproveitar todas as funcionalidades disponíveis.

4.1.3 Solicitação de Serviço

A função de solicitar serviços é uma parte essencial do programa, permitindo que os usuários façam pedidos para que sejam analisados e encaminhados pelo administrador. Essa solicitação pode ser realizada apenas por dois tipos de usuários: prestadores de serviço e solicitantes. Através dessa função, eles podem abrir chamados e pedir serviços de manutenção conforme suas necessidades.

Para que um usuário consiga fazer essa solicitação, é necessário que ele esteja conectado em sua conta de funcionário, garantindo que todas as funcionalidades do sistema estejam operando corretamente.

Quando o usuário estiver conectado e desejar solicitar um serviço, ele deve fornecer algumas informações ao sistema, como: o tipo de serviço que precisa ser realizado, a urgência da tarefa e a especialidade do profissional necessário para resolver o problema.

Se a solicitação for enviada corretamente, o programa deve exibir uma mensagem confirmando que o pedido foi realizado. Em seguida, o usuário será redirecionado para a tela de requisição de serviço, onde poderá abrir outra solicitação ou, se preferir, retornar à página inicial do programa.

4.1.4 Recebimento de solicitação de serviço

A função de recebimento de solicitação de serviço é uma ordem de prioridade considerada essencial para o programa. O recebimento de solicitação de serviço consiste em permitir que o administrador receba um requerimento de serviço, isto é, o chamado de serviço que precisará ser realizado, para que assim, o mesmo analise

as descrições informadas e toda a situação exposta e encaminhe o serviço para o funcionário o qual deverá realizá-lo.

Para que essa ordem possa ser efetuada o administrador deve estar conectado em sua respectiva conta para que assim possa executar esta função pois essa etapa só ficará disponível para a conta do administrador, e também é necessário que uma solicitação de serviço tenha sido enviada anteriormente.

Todos os tipos de usuários poderão visualizar as solicitações de serviços após a realização dessa tarefa.

4.1.5 Designação do funcionário

A designação de funcionário é uma função fundamental dentro do programa, relacionada ao recebimento de solicitações de serviço. Essa função permite que o administrador atribua um funcionário específico para atender a cada pedido, levando em consideração a área de especialização do funcionário e a prioridade da solicitação.

Para que o administrador possa realizar essa tarefa, é necessário que ele esteja conectado em sua conta e que já tenha recebido uma solicitação de serviço previamente.

Quando o administrador está conectado e há uma nova solicitação, ele deve selecioná-la e designar o funcionário mais adequado para realizar o serviço. Além disso, o administrador deve definir a prioridade da tarefa, que pode ser classificada como baixa, média ou alta urgência. Após completar essas etapas, a solicitação é enviada, tornando-a visível para todos os usuários e permitindo que o funcionário designado receba e execute a tarefa.

Se a solicitação for enviada corretamente, o sistema exibirá uma mensagem confirmando que a tarefa foi entregue ao funcionário. Em seguida, o administrador será redirecionado para a página de recebimento de solicitações de serviço.

4.1.6 Início do Serviço

A função de início de serviço é fundamental para o funcionamento do programa, pois permite que o prestador de serviço responda aos chamados e execute as tarefas que lhe foram atribuídas. Para que essa função seja realizada, o prestador de serviço deve estar conectado em sua conta e acessar a área onde estão listados os serviços pendentes. É importante que, previamente, o administrador tenha enviado e designado um chamado ao funcionário.

Na seção "Meus serviços", caso haja uma solicitação pendente, o funcionário deve clicar no botão para visualizar os detalhes do serviço e, em seguida, aceitar a tarefa. Após essa ação, o sistema exibirá uma mensagem de confirmação, indicando que o funcionário iniciou a execução da tarefa. Nesse momento, o serviço passará a ser exibido como "Serviço em aberto" na tela de chamados pendentes. O funcionário, então, poderá dar início à execução do serviço, conforme as instruções fornecidas pelo sistema.

Mesmo após o serviço estar marcado como "Serviço em aberto", o funcionário pode acessar novamente os detalhes do chamado e terá a opção de finalizar o serviço quando concluir a tarefa.

4.1.7 Confirmação da execução do Serviço

Se a confirmação for enviada corretamente, o sistema exibirá uma mensagem confirmando. A função de confirmação de execução de serviço é um processo fundamental para o programa, pois permite que o funcionário prestador de serviço registre que uma tarefa foi realizada e finalizada.

Para que essa confirmação seja possível, o funcionário deve ter concluído o serviço anteriormente.

Caso o prestador deseje confirmar a execução do serviço, ele deve acessar a página de confirmação em sua conta e preencher os dados solicitados pelo sistema. Esses dados incluem informações sobre o status da tarefa, que pode ser: se a tarefa foi completada, se foi realizada parcialmente e necessita de mais cuidados, ou se não foi possível realizá-la. Além disso, há um campo para adicionar informações adicionais, se necessário.

Se a confirmação for enviada corretamente, o sistema exibirá uma mensagem confirmando que a confirmação foi enviada com sucesso. Em seguida, o usuário será redirecionado de volta para a página de confirmação de execução de serviço.

4.1.8 Finalização de Serviço no Sistema

A função de finalização de serviço é um aspecto crucial do programa, permitindo que o administrador conclua uma ordem de serviço após o prestador de serviço confirmar que a tarefa foi realizada.

Para que essa função seja executada, é necessário que o prestador de serviço tenha enviado uma confirmação de execução de serviço ao administrador.

Quando o administrador clicar no botão para encerrar o chamado, o sistema exibirá uma mensagem confirmando que o serviço foi finalizado. Se a finalização for realizada com sucesso, um relatório será gerado automaticamente após a exibição da mensagem de confirmação.

4.1.9 Criação de um Relatório do Serviço realizado

A elaboração de um relatório do serviço realizado é uma etapa do sistema que envolve a coleta de dados sobre o serviço finalizado, permitindo que o supervisor de manutenção elabore a ordem de serviço. Para que isso seja possível, é necessário que o administrador tenha finalizado o chamado no sistema.

Assim que o administrador encerra o chamado, o sistema gera automaticamente um relatório contendo todas as informações relevantes, como o local, horário, tipo de serviço, importância do chamado, entre outros dados que serão exibidos na tela do colaborador. Essas informações auxiliam o supervisor na criação da ordem de serviço, que deve ser registrada no SAP, o sistema utilizado pelo SENAI para gerenciar as ordens de serviço da unidade escolar de Araraquara.

O administrador tem a opção de baixar o arquivo do relatório, se desejar. Após a criação da ordem de serviço, ele pode retornar à tela de finalização de serviço. Caso precise visualizar o relatório novamente, ele estará disponível no histórico do programa, acessível pela opção "Ver mais", onde o documento fica armazenado.

4.1.10 Histórico de Serviços Realizado

O histórico de serviços realizados é uma função fundamental do programa, pois permite que o supervisor de manutenção visualize todos os serviços que já foram executados.

Para acessar esse histórico, o usuário precisa estar conectado em sua conta de administrador. Além disso, é necessário que pelo menos um serviço tenha sido concluído e verificado pelo colaborador responsável.

Uma vez acessado, o histórico será exibido de forma organizada, apresentando informações como: tipo de serviço realizado, local, funcionário responsável, data e horário da conclusão. Isso facilita o acompanhamento e a análise dos serviços prestados.

4.1.11 Criptografia de Senha

Esse requisito define a forma como as senhas dos usuários serão armazenadas e protegidas no sistema, garantindo a segurança das informações pessoais.

Quando o usuário preenche o campo ‘senha’ na tela de cadastro do sistema, os dados deste campo passam por um processo de criptografia onde os caracteres se transformam em código binário - representação de dados utilizando apenas 0 e 1- em que os computadores funcionam internamente com esse formato. Graças a este processo o algoritmo SHA-256 se apropria destes dados para transformá-lo em uma string hexadecimal - representação dos dados em números de 0 a 9 e letras de A a F - tornando-o mais fácil de ler e armazenar no banco de dados. O Algoritmo SHA-256 é unidirecional, ou seja, uma vez que usado para criptografar a senha é

praticamente impossível reverter o processo e obter os dados originais digitados pelo usuário. Isto garante a segurança das informações inseridas no sistema

4.1.12 Facilidade de Uso

Refere-se à experiência do usuário com o sistema, assegurando que ele seja intuitivo e simples de utilizar, mesmo para usuários com pouca experiência técnica.

A interface visual do projeto foi construída com o intuito de auxiliar a melhor navegação do usuário entre as telas do sistema, ajudando na seleção dos serviços encaminhados, na visualização e compreensão das solicitações iniciadas e finalizadas. O posicionamento dos botões com cores chamativas em áreas específicas; tamanho das fontes dos títulos e descrições de serviço; elementos interativos com efeitos; ícones facilitadores para o acesso a outras páginas - em específico a tela do administrador- e mensagens de aviso para direcionar o funcionário técnico ou solicitante. Portanto, os elementos que compõem o visual do sistema contribuem para a facilitação de usabilidade do usuário.

4.1.13 Rápido Acesso

Trata da performance do sistema em termos de resposta rápida às interações do usuário, como tempo de carregamento de páginas ou processamento de informações.

O sistema oferece um acesso rápido e eficiente por meio da utilização de AJAX e Fetch - ferramenta que permite que o sistema troque informações com os servidores em segundo plano e tragam dados de maneira rápida e eficiente -, que permitem a comunicação assíncrona entre a interface do usuário e o servidor. Isso significa que as informações podem ser enviadas e recebidas sem a necessidade de

recarregar a página, proporcionando uma navegação fluida e sem interrupções. Além disso, a estrutura otimizada das consultas SQL no back-end garante que os dados sejam acessados e processados de forma eficiente, minimizando o tempo de resposta do sistema. As requisições são realizadas de maneira direta e rápida, sem sobrecarregar o servidor, o que resulta em uma experiência de usuário ágil e com um tempo de resposta extremamente reduzido. Isso permite que os solicitantes, técnicos de manutenção e administrador solicitem, acompanhem e visualizem as solicitações de serviço de forma imediata.

4.1.14 Multiplataforma

Este requisito refere-se à capacidade do sistema de funcionar corretamente em diferentes dispositivos e sistemas operacionais, garantindo a acessibilidade para todos os tipos de usuários

A acessibilidade em diversos dispositivos garante que os usuários; solicitantes, técnicos de manutenção e administrador, possam acessar o sistema de computadores, smartphones ou tablets, em qualquer lugar e a qualquer momento. Além disso, a compatibilidade com diferentes sistemas operacionais assegura que o sistema funcione em plataformas como Windows, macOS, Android e iOS, permitindo que todos os usuários, independentemente do dispositivo, utilizem o sistema possibilitando-os de visualizar e solicitar serviços. Bem como a interface responsiva e adaptável que se ajusta automaticamente ao tamanho da tela, proporcionando uma experiência de uso otimizada, seja em dispositivos móveis ou desktop.

5 CONCLUSÃO

O presente estudo buscou criar um sistema que pudesse organizar as demandas de solicitações de serviço da manutenção do SENAI, e os resultados obtidos evidenciam que o sistema desenvolvido atendeu às expectativas e necessidades.

Após a finalização do protótipo e os testes realizados, foi possível atestar que o sistema realmente atende os objetivos e que, se implementado em um ambiente com muitas pessoas, surgirão benefícios reais para todos os usuários. Ao organizar a alta demanda de serviços a serem realizados, o sistema contribui significativamente para a segurança do ambiente em que for implementado. Os testes do protótipo foram realizados pelo supervisor de manutenção do SENAI "Henrique Lupo" de Araraquara, com o principal objetivo de avaliar as funcionalidades principais do sistema como: envio e encaminhamento de solicitações de serviço e design das telas. Após os testes, o supervisor de manutenção respondeu um formulário, com perguntas sobre as funcionalidades e o design. O teste e as respostas mostraram que o sistema atende seus objetivos e que é uma ferramenta muito útil para o setor de manutenção. Durante o desenvolvimento do projeto, o grupo teve a oportunidade de aprender a planejar e desenvolver um sistema, desde a criação de diagramas por meio de ferramentas digitais até o início da codificação. Além disso, desenvolvemos muitas habilidades socio-emocionais, como: Paciência, Organização, Responsabilidade, Auto Controle, Trabalho em Equipe e Tomada de Decisões. Por fim, uma possível melhoria para o sistema seria a criação de uma página que possibilitaria o monitoramento dos serviços realizados por meio de gráficos, para assim facilitar o gerenciamento das manutenções preventivas.

REFERÊNCIAS

SWEETALERT2. **SweetAlert2**. *SweetAlert2*. Disponível em: <https://sweetalert2.github.io/>. Acesso em: 10 set. 2024.

W3SCHOOLS**. W3Schools Online Web Tutorials**. *W3Schools*. Disponível em: <https://www.w3schools.com/>. Acesso em: 15 set. 2024.

UX REPUBLIC. **Como criar um modal otimizado**. *UX Republic*. Disponível em: <https://www.ux-republic.com/pt/como-criar-um-modal-otimizado/>. Acesso em: 29 set. 2024.

GETBOOTSTRAP. **Modal**. *Bootstrap*. Disponível em: <https://getbootstrap.com/docs/4.0/components/modal/>. Acesso em: 05 out. 2024.

CASA DO DESENVOLVEDOR. **Requisitos funcionais e não funcionais**. *Blog Casa do Desenvolvedor*. Disponível em: <https://blog.casadodesenvolvedor.com.br/requisitos-funcionais-e-nao-funcionais/>. Acesso em: 18 out. 2024.

W3SCHOOLS. **How To Custom Select**. *W3Schools*. Disponível em: <https://www.w3schools.com/howto/howto_custom_select.asp>. Acesso em: 18 out. 2024.

ALURA. **Autenticação de forma segura com criptografia**. *Alura*. Disponível em: <https://www.alura.com.br/artigos/autenticacao-de-forma-segura-com-criptografia?srsltid=AfmBOoo8BVu6faAjCFPYi31_dGMm5Fitssheh-EMeqmOgqPBq68jyGW7>. Acesso em: 18 out. 2023.

MOZILLA FOUNDATION. **CSS transform function: scale**. *Mozilla Foundation*. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/scale>. Acesso em: 24 out. 2024.

https://www.produttivo.com.br/blog/tipos-de-manutencao-quais-suas-diferencas/ acesso em 13/09/2024

https://developer.mozilla.org/pt-BR/docs/Web/CSS/word-break acesso em 20/11/2024

Apêndice

Apêndice A – Tela inicial do EASY-REQUEST

Este apêndice contém os códigos referentes a implantação da Tela Inicial do Easy Request. A Tela Inicial inclui a logo do Easy Request e textos que direcionam as ações que o usuário deve tomar com relação aos botões “Entrar” e “Cadastrar-se”.

Na Tela Inicial, o HTML contém um cabeçalho com a logo do Easy Request e um container principal com os textos que direcionam ao usuário e os botões de “Entrar” e “Cadastrar-se”.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <!-- Link do CSS -->
    <link rel="stylesheet" href="/static/CSS/global.css">
    <link rel="stylesheet" href="/static/CSS/index.css">
    <!-- Link JS da pagina -->
    <script defer src="/static/JS/index.js"></script>
    <!-- Logo do Easy Request na guia-->
    <link rel="shortcut icon" type="imagex/png"
href="/static/IMG/TASK request.svg">
    <!-- link JS biblioteca scrollrevealjs -->
    <script src="https://unpkg.com/scrollreveal"></script>
    <title>EASY-REQUEST</title>
</head>
<body>
    <!-- Navegação principal do site -->
    <nav>
        <div class="nav-img">
            <!-- Logo -->
            <img id="logo" src="/static/IMG/Brand Identity and
Design.svg" alt="Logotipo do Easy-Request">
        </div>
    </nav>
    <main>
        <!-- Container principal para o conteúdo da página -->
        <div class="container-abaixo" id="container-abaixo">
```

```
                <div class="container__abaixo--conteudotxt">
                    <!-- Título e descrição da página -->
                    <h2>BEM-VINDO AO</h2>
                    <h2>EASY-    <span    class="container__abaixo--
txtVazado">REQUEST</span></h2>
                    <h3                  class="container__abaixo--
descricao">Gerencie seus chamados de forma simples e rápida</h3>
                </div>
                <div class="container__abaixo--botoes">
                    <!-- Mensagem para usuários existentes e novos
-->
                    <p>Você já possui Cadastro?</p>
                    <!-- Botão para login -->
                    <a     href="/RF002"     class="botao     botao-
login">Entrar</a>
                    <p>ou</p>
                    <!-- Botão para cadastro -->
                    <a     href="/RF001"     class="botao     botao-
cadastro">Cadastre-se</a>
                </div>
            </div>
        </main>
    </body>
    </html>
```

Na Tela Inicial, o CSS é utilizado para estilizar e posicionar os elementos HTML. Os principais elementos estilizados são o container que contém a logo e também o container com os textos e os botões.

```
    ::-webkit-scrollbar {
        width: 0; /* Para navegadores baseados em WebKit */
    }

    body {
        scrollbar-width: none; /* Para Firefox */
        -ms-overflow-style: none; /* Para Internet Explorer e Edge
*/
    }
    /* Container nav */
    nav {
        display: flex;
        justify-content: center;
        align-items: center;
        padding: 1rem;
    }

    /* Estilo para a imagem dentro da nav */
```

```
.nav-img img {
    max-width: 20rem;
    width: 100%;
    height: auto;
}

/* Container principal na parte inferior da tela */
.container-abaixo{
    width: 100vw;
    min-height: 70vh;
    max-height: 70vh;
    overflow-y: auto; /* Adiciona rolagem vertical */
    overflow-x: hidden; /* Esconde a rolagem horizontal */
    background-color: var(--color06);
    margin-bottom: 0;
    position: absolute;
    bottom: 0;
    left: 0;

    border-top-left-radius: 2rem;
    border-top-right-radius: 2rem;
    padding: 1.5rem;

    display: flex;
    flex-direction: column;
    align-content: center;
    flex-wrap: nowrap;
    justify-content: space-evenly;
    box-shadow: -2px -1px 10px rgba(0, 0, 0, 0.5);
}

/* Texto centralizado dentro do container */
.container__abaixo--conteudotxt{
    width: 100%;
    text-align: center;
    font-size: 2.4rem;
    margin-top: 3rem;
}

/* Estilo do título dentro do texto */
.container__abaixo--conteudotxt h2{
    color: var(--color01);
}

/* Descrição com estilo específico */
.container__abaixo--descricao{
    font-size: 2rem;
    margin-top: 2rem;
    color: var(--color01);
```

```
}

/* Cor do TXT vazado (REQUEST) */
.container__abaixo--txtVazado{
    color: var(--color06);
    text-shadow:
      1.5px 1.5px 0 #ffffff,
      -1.5px 1.5px 0 #ffffff,
      -1.5px -1.5px 0 #ffffff,
      1.5px -1.5px 0 #ffffff;
}

/* Container para os botões na parte inferior */
.container__abaixo--botoes {
    text-align: center;
    margin-top: 3rem;
}

/* Estilos dos botões */
.botao {
    display: inline-block;
    width: 30rem;
    padding: 1.2rem;
    text-align: center;
    border-radius: 50px;
    font-size: 1.6rem;
    font-weight: bold;
    text-decoration: none;
    transition: all 0.3s ease-in-out;
    cursor: pointer;
}

/* Estilo específico para o botão de login */
.botao-login {
    background-color: var(--color03);
    border: 2px solid var(--color03);
    color: var(--color06);
    margin-bottom: 1rem;
}

/* Estilo do botão de login ao passar o mouse */
.botao-login:hover {
    background-color: transparent;
    color: var(--color03);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.05);
}

/* Estilo específico para o botão de cadastro */
```

```
    .botao-cadastro {
        background-color: transparent;
        border: 2px solid var(--color03);
        color: var(--color03);
    }

    /* Estilo do botão de cadastro ao passar o mouse */
    .botao-cadastro:hover {
        background-color: var(--color03);
        color: var(--color06);
        box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
        transform: scale(1.05);
    }

    /* Estilo do texto dentro do container de botões */
    .container__abaixo--botoes p {
        font-size: 1.6rem;
        color: var(--color01);
        margin: 1rem 0;
    }
    /* ----------------------EFEITO DE TRANSIÇÃO-----------------
*/
    #logo{
        visibility: hidden;
    }
    #container-abaixo{
        visibility: hidden;
    }
```

O JavaScript da Tela Inicial tem como principal objetivo criar uma animação no momento do carregamento da página, por meio de uma biblioteca chamada ScrollReveal.

```
    // Inicializa o ScrollReveal e define a configuração global
    window.sr = ScrollReveal({
        // Define se a animação será executada novamente quando o
elemento reaparecer na visualização. 'true' significa que o efeito
se repetirá sempre que o elemento entrar na tela.
        reset: true
    });

    // Aplica o efeito de revelação ao elemento com o ID 'logo'
    sr.reveal('#logo', {
```

```
        duration: 500,
        origin: 'top',
        distance: '50px'
    });

    // Aplica o efeito de revelação ao elemento com o ID
'container-abaixo'
    sr.reveal('#container-abaixo', {
        duration: 500,
        origin: 'bottom',
        distance: '50px',
        easing: 'ease-out'
    });
```

No BackEnd, o Python renderiza a página e adiciona o nome e função do usuário no HTML utilizando a rota “/”. Já a rota “/firebase-messaging-sw.js” serve para carregar o arquivo “firebase-messaging-sw.js” de uma forma que o Flask possa lê-lo normalmente. No início do arquivo é necessário fazer todas as importações para garantir o funcionamento adequado da aplicação:

- Flask e suas extensões: Importamos o Flask para conseguir manipular as rotas e iniciar o servidor;
- os: Importamos o “os” para conseguir executar o arquivo “firebase-messaging.js” de forma correta, mesmo ele estando na raiz do projeto;
- firebase_admin e suas extensões: Importamos o “firebase_admin” utilizado para criar as funcionalidades das notificações;
- conexao_SQL: Importamos a classe “conexao_SQL” para podermos conectar o Python com o Banco de Dados MySQL.;
- Usuario: Importamos a classe “Usuario” para conseguirmos realizar todas as ações relacionadas ao usuário, como cadastro e login;
- Solicitacao: Importamos a classe “Solicitacao” para conseguirmos realizar todas as ações relacionadas às solicitações de serviço, como enviar uma solicitação;

- Encaminhamento: Importamos a classe “Encaminhamento” para conseguirmos realizar todas as ações relacionadas aos encaminhamentos de serviço, como encaminhar um serviço;
- update_file: Importamos a função “update_file” para conseguirmos transformar as fotos adicionadas em links para serem adicionadas ao Banco de Dados;

Após as importações, iniciamos o servidor com o código “Flask(__name__)”, criamos uma chave de segurança com o código “app.secret_key” e depois iniciamos o Firebase Admin SDK com a chave privada.

Depois, com a rota “/firebase-messaging-sw”, executamos o arquivo “firebase-messaging-sw.js” de forma correta, evitando conflitos com o Flask. Já a rota “/” renderiza a Tela Inicial.

```
    # Importando o Flask e componentes importantes para o código
    from flask import Flask, render_template, request, redirect,
session, jsonify, send_from_directory

    # Importando os componentes do Firebase
    import os
    import firebase_admin
    from firebase_admin import credentials, messaging

    # Importando Classes dos arquivos autorais
    from conexao_SQL import Connection
    from Usuario import Usuario
    from Solicitacao import Solicitacao
    from Encaminhamento import Encaminhamento

    # Importando função para salvar fotos como links
    from upload_file import upload_file

    # Instanciando o WebService
    app = Flask(__name__)

    # Criando uma senha para criptografar sessão
    app.secret_key = "capivara"

    # Inicializar o Firebase Admin SDK com a chave privada
```

```
    cred = credentials.Certificate("./easy-request-17b8a-firebase-adminsdk-kwd4p-20f87b1f6b.json")
    firebase_admin.initialize_app(cred)

    # Criando a rota para renderizar o arquivo 'firebase-messaging-sw.js' como arquivo static
    @app.route('/firebase-messaging-sw.js')
    def sw(): # Função que renderiza o arquivo 'firebase-messaging-sw.js' que está na raiz do projeto, como se ele estivesse na pasta static
        return send_from_directory(os.path.join(app.root_path), 'firebase-messaging-sw.js')

    # Criando a rota para a página inicial
    @app.route("/")
    def pg_inicio(): # Função que renderiza a tela inicial
        if "usuario" in session:
            permissao = session["usuario"]["permissao"]

            if permissao == "administrador":
                return redirect("/tl-administrador")
            elif permissao == "manutencao":
                return redirect("/RF006")
            elif permissao == "solicitante":
                return redirect("/tl-solicitante")
        else:
            return render_template("index.html")
```

Apêndice B – Tela Cadastro do EASY REQUEST

Este apêndice contém os códigos referentes a Tela de Cadastro do Easy Request. A Tela de Cadastro inclui um formulário que coleta as principais informações do usuário, que são: cargo, nome, CPF, SN, e-mail e senha. O Cadastro do usuário é efetuado por uma junção de códigos JavaScript e Python.

No HTML da Tela de Cadastro existe uma seção dentro do conteúdo principal da página(main) que é o formulário. Dentro da seção do formulário existem vários containers que contém os campos do formulário onde são inseridas as informações.

```
    <!DOCTYPE html>
    <html lang="pt-br">
    <head>
```

```
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <!-- Link do CSS da pagina -->
        <link rel="stylesheet" href="/static/CSS/global.css">
        <link rel="stylesheet" href="/static/CSS/RF001-cad.css">
        <!-- Link do JS da pagina -->
        <script src="/static/JS/RF001-cad.js" defer></script>
        <!--  Biblioteca  jQuery  para  facilitar  o  uso  de  Ajax  e
manipulação de DOM -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <!-- link JS biblioteca scrollrevealjs -->
        <script src="https://unpkg.com/scrollreveal"></script>
        <!-- Logo do Easy Request na guia-->
        <link      rel="shortcut      icon"      type="imagex/png"
href="/static/IMG/TASK request.svg">
        <title>Cadastro</title>
    </head>
    <body>
        <!-- Navegação principal do site -->
        <main class="main">
            <section id="form-container" class="form-container ">
                <h2 class="form-container__title">Cadastro</h2>

                    <!-- Seção de seleção de profissão -->
                    <div class="form-container__input">
                        <!-- Rótulo para a seleção da profissão -->
                        <label      for="funcao"      class="form-
container__label">Informe seu Cargo:</label>
                        <!--  Campo  de  seleção  com  opções  de
profissão -->
                        <select      id="funcao"      name="funcao"
class="form-container__select">
                            <option value="">Selecione...</option>
                            <option      value="1">Técnico      de
Manutenção</option>
                            <option     value="2">Assistente     de
Serviços Administrativos</option>
                            <option              value="3">Auxiliar
Técnico</option>
                            <option      value="4">Ajudante      de
Expediente</option>
                            <option value="5">Analista de Qualidade
de Vida</option>
                            <option
value="6">Blibliotecário(a)</option>
```

```
                                <option     value="7">Coordenador     de
Atividades Técnicas</option>
                                <option           value="8">Coordenador
Pedagógico</option>
                                <option value="9">Diretor</option>
                                <option
value="10">Especialista</option>
                                <option              value="11">Gerente
Administrativo e Financeiro</option>
                                <option
value="12">Instrutor(a)</option>
                                <option      value="13">Oficial      de
Manutenção</option>
                                <option value="14">OPP</option>
                                <option     value="15">Orientador     de
Prática Profissional</option>
                                <option value="16">Outros</option>
                            </select>
                        </div>

                        <!-- Campo de entrada para o nome completo -->
                        <div class="form-container__input">
                            <label        for="nome"        class="form-
container__label">Nome completo:</label>
                            <input    autocomplete="off"    type="text"
id="nome"  name="nome"  class="form-container__input-text  txt-nome"
placeholder="Digite seu nome:" required>
                        </div>

                        <!-- Campo de entrada para o CPF -->
                        <div class="form-container__input">
                            <label        for="cpf"        class="form-
container__label">CPF:</label>
                            <input    autocomplete="off"    type="text"
id="cpf"         name="cpf"         class="form-container__input-text"
placeholder="000.000.000-00" maxlength="14" required>
                        </div>

                        <!-- Campo de entrada para o SN -->
                        <div class="form-container__input">
                            <label         for="sn"         class="form-
container__label">SN:</label>
                            <input    autocomplete="off"    type="text"
id="sn"          name="sn"          class="form-container__input-text"
placeholder="SN0000000" required>
                        </div>

                        <!-- Campo de entrada para o e-mail -->
                        <div class="form-container__input">
```

```
                        <label       for="email"       class="form-
container__label">E-mail:</label>
                        <input    autocomplete="off"    type="email"
id="email"   name="email"   class="form-container__input-text   email"
placeholder="exemplo@exemplo.com" required>
                    </div>

                    <!-- Campo de entrada para a senha -->
                    <div class="form-container__input">
                        <label       for="senha"       class="form-
container__label">Senha:</label>
                        <div class="senha-container">
                            <input    type="password"    id="senha"
name="senha"         class="form-container__input-text         senha"
placeholder="Pelo menos 6 caracteres" required>
                            <button  type="button"  id="toggleSenha"
class="form-container__toggle-button">
                                <img              id="cadeadoIcone"
src="/static/IMG/fechar-o-olho.png" alt="Mostrar senha" />
                            </button>
                        </div>
                        <!-- Validação a senha -->
                        <div   id="senha-requisitos"   class="senha-
requisitos hidden">
                            <p                    id="requisito1"
class="requisito">Mínimo de 6 caracteres</p>
                            <p                    id="requisito2"
class="requisito">Uma letra</p>
                            <p id="requisito3" class="requisito">Um
número</p>
                            <p id="requisito4" class="requisito">Um
caractere especial</p>
                            <p     id="mensagem"     class="mensagem
hidden">Senha validada</p>
                        </div>
                    </div>

                    <!-- Campo de upload de foto -->
                    <div class="form-container__input">
                        <label       for="foto"       class="form-
container__label">Foto:</label>
                        <input  type="file"  id="foto"  name="foto"
class="form-container__input-file">
                    </div>

                    <!-- Botão de submissão do formulário -->
                    <button          class="form-container__button"
onclick="cadastrar()">Cadastrar-se</button>
            </section>
```

```
        </main>
    </body>
    </html>
```

Na Tela de Cadastro, o CSS é utilizado para estilizar o formulário e todos os seus campos, para tornar a página mais atraente e intuitiva. Ele também é responsável por permitir que a página se adapte a diferentes tipos de tela por meio da regra “@media”, que define algumas alterações de layout de acordo com a largura da tela.

```
    html {
        font-size:70.5%;

    }
    /* Classe para esconder elementos quando necessário */
    .hidden {
        display: none;
    }

    body {
        overflow-x: hidden;
        scrollbar-width: none;
        -ms-overflow-style: none;
    }
    ::-webkit-scrollbar {
        width: 0;
    }

    /* Container principal do formulário, define o layout da página
*/
    .form-container {
        width: 100vw;
        height: auto;
        padding: 3.6rem;
        background-color: var(--color06);
        color: var(--color01);
        border-bottom-left-radius: 2rem;
        border-bottom-right-radius: 2rem;
        overflow: hidden;
        box-shadow: -2px 5px 10px rgba(0, 0, 0, 0.5);
    }
    /* Estilo do título do formulário */
    .form-container__title {
        font-size: 3.3rem;
        text-align: center;
        margin-bottom: 1rem;
```

```
}

/* Estilo do campo de seleção (dropdown) */
.form-container__select {
    width: 100%;
    padding: 0.5rem;
    border-radius: 5px;
    border: 1px solid var(--color03);
}

/* Container das imagens, define o alinhamento */
.form-container__images {
    display: flex;
    width: 110px;
    margin: 0 auto;
    justify-content: center;
    align-items: center;
    flex-direction: row;
}

/* Estilo das imagens dentro do formulário */
.form-container__image {
    max-width: 100px;
    margin: 0 10px;
}

/* Container para os campos de input */
.form-container__input {
    margin-bottom: 1rem;
}

/* Estilo dos rótulos dos campos */
.form-container__label {
    color: var(--color03);
    font-size: 1.3rem;
}


/* Estilo dos campos de texto do formulário */
.form-container__input-text, .form-container__input-file{
    width: 100%;
    padding: 0.5rem;
    border-radius: 5px;
    border: 1px solid var(--color03);

}
.txt-nome{
    text-transform: capitalize;
}
```

```css
/* Estilo do campo de upload de arquivos */
.form-container__input-file {
    padding: 5px;
}

/* CSS do btn */
.form-container__button {
    width: 100%;
    background-color: var(--color03);
    border: 2px solid var(--color03);
    color: var(--color06);
    margin-bottom: 1rem;
    padding: 10px;
    border-radius: 5px;
    cursor: pointer;
    box-sizing: border-box;
}

/* Estilo do botão ao passar o mouse (hover) */
.form-container__button:hover {
    background-color: transparent;
    color: var(--color03);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.02);
    border: 2px solid var(--color03);
}
/* Estilo dos campos de texto do formulário */
.form-container__input-text,
.form-container__input-file,
.form-container__select,
.senha {
    width: 100%;
    padding: 0.5rem;
    border: none;
    border-bottom: 2px solid var(--color03);
    background-color: transparent;
    outline: none;
}

/* Adiciona um efeito de foco para os inputs */
.form-container__input-text:focus,
.form-container__input-file:focus,
.form-container__select:focus {
    border-bottom: 2px solid var(--color01);
}
option{
    color: black;
    background-color: rgba(19, 43, 46, 0.332);
    border-bottom-left-radius: 2rem;
```

```
        border-bottom-right-radius: 2rem;
    }
    /* Estilo dos campos de texto do formulário */
    .form-container__input-text,
    .form-container__input-file,
    .form-container__select {
        width: 100%;
        padding: 0.5rem;
        border: none; /* Remove todas as bordas */
        border-bottom: 2px solid var(--color03);
        background-color: transparent;
        outline: none;
        color: white;
    }

    /* Muda a cor do placeholder para branco */
    .form-container__input-text::placeholder,
    .form-container__input-file::placeholder,
    .form-container__select::placeholder {
        color: white;
        opacity: 1;
    }

    /* Para garantir a cor do texto no select */
    .form-container__select {
        color: white;
        background-color: transparent;
    }
    /* Estilo para inputs preenchidos automaticamente */
    .form-container__input-text:-webkit-autofill,
    .form-container__input-file:-webkit-autofill,
    .form-container__select:-webkit-autofill {
        -webkit-box-shadow: 0 0 0 1000px transparent inset;
        box-shadow: 0 0 0 1000px transparent inset;
        color: white;
        background-color: transparent !important;
    }

    /* Ideias de teste */
    .senha-requisitos {
        margin-top: 0.5rem;
    }

    .requisito {
        color: red;
        font-size: 1.1rem;
    }

    .requisito.validado {
```

```
        color: green;
    }

    .mensagem {
        margin-top: 0.5rem;
        color: green;
    }

    .senha-container {
        display: flex;
        align-items: center;
    }

    .form-container__toggle-button {
        background: none;
        border: none;
        cursor: pointer;
        padding: 0;
    }

    .form-container__toggle-button img {
        width: 24px;
        height: 24px;
        margin-left: 5px;
    }

    /* ------------------------ MEDIA ------------------------ */
    @media (min-width: 768px) {
        html {
            font-size:75.5%;

        }
        body{
            width: 100vw;
            height: 100vh;
            background: url(/static/IMG/Telacad.png);
            background-repeat: no-repeat;
            background-size: cover;

        }
        .form-container {
            width: 100vw;
            max-width: 800px;
            padding: 2rem;
            margin: 0 auto;
        }

        .form-container__input {
            margin-bottom: 0.8rem;
```

```
        }

        .form-container__input-text, .form-container__input-file {
            padding: 1rem;
            font-size: 0.9rem;
        }

        .form-container__select {
            padding: 1rem;
        }

        .form-container__button {
            padding: 8px;
            font-size: 1.6rem;
            margin-bottom: 3rem;
        }

        .form-container__button:hover {
            transform: scale(1.02);
        }
    }

    /* Media Query para telas (acima de 1200px) */
    @media (min-width: 1200px) {
        html {
            font-size: 72.5%;

        }
        body{
            background: url(/static/IMG/Telacad.png);
            background-repeat: no-repeat;
            background-size: cover;

        }
        .form-container {
            width: 70vw;
            max-width: 1000px;
            padding: 2.4rem;

        }

        .form-container__input {
            display: flex;
            flex-direction: column;
            margin-bottom: 1rem;

        }

        .form-container__label {
```

```
            margin-bottom: 0.5rem;
        }

        .form-container__input-text, .form-container__input-file {
            padding: 1rem;
            font-size: 1rem;
        }

        .form-container__select {

            padding: 1rem;
        }
        .form-container__button {
            padding: 12px;
            font-size: 1.5rem;
            margin-bottom: 3rem;
        }

        .form-container__button:hover {
            transform: scale(1.02);
        }
    }
```

O JavaScript da Tela de Cadastro tem como principal função enviar as informações coletadas pelo formulário para o Python por meio da função “cadastrar()”. Além disso, o JavaScript formata os campos nome, CPF e SN automaticamente, facilitando o uso das informações posteriormente. O campo de senha também possui uma verificação automática, que ajuda o usuário a criar uma senha segura.

```
    // Inicializa o ScrollReveal e define a configuração global
    window.sr = ScrollReveal({
        // Define se a animação será executada novamente quando o elemento reaparecer na visualização. 'true' significa que o efeito se repetirá sempre que o elemento entrar na tela.
        reset: true
    });

    // Aplica o efeito de revelação ao elemento com o ID '#form-container'
    sr.reveal('#form-container', {
        duration: 500,
        origin: 'top',
        distance: '50px'
```

```
    });

    // Função para o campo de CPF
    document.getElementById('cpf').addEventListener('input',
function (e) {
        let cpf = e.target.value.replace(/\D/g, ''); // Remove tudo
que não for número
        cpf = cpf.replace(/(\d{3})(\d)/, '$1.$2'); // Adiciona o
primeiro ponto
        cpf = cpf.replace(/(\d{3})(\d)/, '$1.$2'); // Adiciona o
segundo ponto
        cpf   =   cpf.replace(/(\d{3})(\d{1,2})$/,   '$1-$2');   //
Adiciona o traço

        e.target.value = cpf;
    });

    // Função para o campo de SN
    // Prefixo fixo
    const prefixo = "SN";
    const snInput = document.getElementById('sn');

    // Adiciona o prefixo no campo quando o usuário clica no campo
    snInput.addEventListener('focus', function(){
        if (snInput.value === '') {
            snInput.value = prefixo;
        }else{
            return;
        }
    })

    // Remove o prefixo no campo quando o usuário clica fora do
campo
    snInput.addEventListener('blur', function(){
        if (snInput.value === prefixo) {
            snInput.value = '';
        }else{
            return;
        }
    })

    // Adiciona o evento de input
    snInput.addEventListener('input', function() {
        // Remove o prefixo para evitar duplicação
        let value = snInput.value.replace(prefixo, '');
        // Remove qualquer caractere que não seja número
        value = value.replace(/\D/g, '');
        // Limita a quantidade de números a 7
        value = value.substring(0, 7);
```

```
        // Atualiza o valor do input com o prefixo "SN" e os números digitados
        snInput.value = prefixo + value;
    });

    // Bloqueia o usuário de deletar o prefixo "SN"
    snInput.addEventListener('keydown', function(e) {
        // Previne que o usuário apague o prefixo
        if (snInput.selectionStart < prefixo.length && (e.key === "Backspace" || e.key === "Delete")) {
            e.preventDefault();
        }
    });

    // Pegando os valores dos inputs
    const cpf = document.getElementById('cpf');
    const nome = document.getElementById('nome');
    const email = document.getElementById('email');
    const senha = document.getElementById('senha');
    const sn = document.getElementById('sn');
    const foto = document.getElementById('foto');
    const id_funcao = document.getElementById('funcao');
    var permissao = '';

    // Função para cadastrar o usuário
    function cadastrar(){
        if (cpf.value === '' || nome.value === '' || email.value === '' || senha.value === '' || sn.value === '' || id_funcao.value === '') {
            Swal.fire({
                icon: "error",
                title: "Oops...",
                text: "Preencha todos os campos obrigatórios corretamente!",
                showConfirmButton: false,
                timer: 3500
              });
            return
        }else if((senha.value).length < 6){
            Swal.fire({
                icon: "error",
                title: "Oops...",
                text: "A senha deve ter pelo menos 6 caracteres",
                showConfirmButton: false,
                timer: 3500
              });
            return
        }
```

```
        var dados = new FormData(); // Cria um novo FormData

        // Adiciona os dados ao FormData
        dados.append('cpf', cpf.value);
        dados.append('nome', nome.value);
        dados.append('email', email.value);
        dados.append('senha', senha.value);
        dados.append('sn', sn.value);
        dados.append('foto', foto.files[0]); // Adiciona a foto
        dados.append('id_funcao',   parseInt(id_funcao.value));   //
Adiciona id_funcao

        $.ajax({
            url: '/cadastrar-usuario',
            type: 'POST',
            data: dados,
            contentType: false, // Importante para enviar arquivos
            processData: false, // Não processar os dados
            success: function(){
                if    (parseInt(id_funcao.value)    >=    2    &&
parseInt(id_funcao.value) <= 16){
                    permissao = 'solicitante';
                }
                else if( parseInt(id_funcao.value) == 1){
                    permissao = 'manutencao';
                }

                console.log(parseInt(id_funcao.value))
                console.log(permissao);


                if(permissao == 'manutencao'){
                    window.location.href = '/RF006';
                }

                else if(permissao == 'solicitante'){
                    window.location.href = '/tl-solicitante';
                }

            },
            error: function(){
                Swal.fire({
                    icon: "error",
                    title: "Oops...",
                    text: "Erro ao Cadastrar!",
                    showConfirmButton: false,
                    timer: 3500
                  });
            }
```

```
        });
    }

    // Função para verificação de senha interativa
    $(document).ready(function() {
        const senhaInput = $('#senha');
        const requisitos = {
            minLength: false,
            hasLetter: false,
            hasNumber: false,
            hasSpecialChar: false
        };

        const requisito1 = $('#requisito1');
        const requisito2 = $('#requisito2');
        const requisito3 = $('#requisito3');
        const requisito4 = $('#requisito4');
        const mensagem = $('#mensagem');

        senhaInput.on('focus', function() {
            $('#senha-requisitos').removeClass('hidden');
        });

        senhaInput.on('input', function() {
            const senha = senhaInput.val();
            requisitos.minLength = senha.length >= 6;
            requisitos.hasLetter = /[a-zA-Z]/.test(senha);
            requisitos.hasNumber = /\d/.test(senha);
            requisitos.hasSpecialChar                                    =
/[!@#$%^&*(),.?":{}|<>]/.test(senha);

            // Atualiza os requisitos
            requisito1.toggleClass('validado',
requisitos.minLength);
            requisito2.toggleClass('validado',
requisitos.hasLetter);
            requisito3.toggleClass('validado',
requisitos.hasNumber);
            requisito4.toggleClass('validado',
requisitos.hasSpecialChar);

            // Verifica se todos os requisitos foram atendidos
            if (requisitos.minLength && requisitos.hasLetter &&
requisitos.hasNumber && requisitos.hasSpecialChar) {
                // Limpa os outros textos e mostra apenas a
mensagem de validação
                requisito1.addClass('hidden');
                requisito2.addClass('hidden');
                requisito3.addClass('hidden');
```

```
                requisito4.addClass('hidden');
                mensagem.removeClass('hidden').text('Senha
validada');
            } else {
                // Restaura os textos dos requisitos se a senha não
for válida
                mensagem.addClass('hidden');
                requisito1.removeClass('hidden');
                requisito2.removeClass('hidden');
                requisito3.removeClass('hidden');
                requisito4.removeClass('hidden');
            }
        });
    });

    // Mostrar e ocultar senha
    document.getElementById('toggleSenha').addEventListener('click'
, function () {
        const senhaInput = document.getElementById('senha');
        const                    cadeadoIcone                    =
document.getElementById('cadeadoIcone');

        if (senhaInput.type === 'password') {
            senhaInput.type = 'text';
            cadeadoIcone.src = '/static/IMG/olho-de-perto.png';
            cadeadoIcone.alt = 'Ocultar senha';
        } else {
            senhaInput.type = 'password';
            cadeadoIcone.src = '/static/IMG/fechar-o-olho.png';
            cadeadoIcone.alt = 'Mostrar senha';
        }
    });

    //Executando a função de cadastro sempre que clicar na tecla
Enter em qualquer campo selecionado ou lugar da tela
    cpf.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            cadastrar();
        }
    });

    nome.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            cadastrar();
        }
    });

    email.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
```

```
            cadastrar();
        }
    });

    senha.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            cadastrar();
        }
    });

    sn.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            cadastrar();
        }
    });

    document.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            cadastrar();
        }
    });
```

No BackEnd, o Python renderiza a página de cadastro através da rota “/RF001”, e a rota “/cadastrar-usuario” efetua o cadastro de usuário a partir de uma função chamada “cadastrar()” que está presente no arquivo com as funções referentes ao Usuario e é responsável por adicionar as informações do usuário no Banco de Dados.

```
    # Criando rota para realizar o cadastro
    @app.route("/cadastrar-usuario", methods=["POST"])
    def cadastrarUsuario(): # Função que executa o cadastro
        cpf = request.form.get('cpf')
        nome = request.form.get('nome')
        email = request.form.get('email')
        senha = request.form.get('senha')
        foto = request.files.get('foto')
        sn = request.form.get('sn')
        id_funcao = int(request.form.get('id_funcao'))

        if foto:
            link_arquivo_foto = upload_file(foto)
        else:
            link_arquivo_foto = None
```

```
        if id_funcao >= 2 and id_funcao <= 16:
            permissao = "solicitante"
        elif id_funcao == 1:
            permissao = "manutencao"

        myBD = Connection.conectar()

        mycursor = myBD.cursor()

        mycursor.execute(f"SELECT nome FROM tb_funcoes WHERE
id_funcao = {id_funcao}")

        funcao = mycursor.fetchone()

        usuario = Usuario()

        if usuario.cadastrar(cpf, nome, email, senha, sn,
link_arquivo_foto, permissao, id_funcao):
            session["usuario"] = {"CPF":cpf ,"nome":nome,"sn":sn,
"foto":link_arquivo_foto, "funcao":funcao[0],"permissao":permissao}

            return jsonify({'mensagem':'Cadastro OK'}), 200
        else:
            return {'mensagem':'ERRO'}, 500
```

```
        # Função que cadastra o usuário no Banco de Dados.
        def cadastrar(self, cpf, nome, email, senha, sn, foto,
permissao, id_funcao):
            try:
                myBD = Connection.conectar()

                mycursor = myBD.cursor()

                self.cpf = cpf
                self.nome = nome
                self.email = email
                self.senha = senha
                self.sn = sn
                self.foto = foto
                self.permissao = permissao
                self.id_funcao = id_funcao
                self.logado = True
                senha_criptografada =
sha256(self.senha.encode()).hexdigest()

                if foto:
                    mycursor.execute(f"INSERT INTO tb_funcionarios
(CPF_funcionario, nome, email, senha, SN, foto, permissao,
id_funcao) VALUES ('{cpf}','{nome}', '{email}',
```

```
'{senha_criptografada}',     '{sn}',     '{foto}',     '{permissao}',
{id_funcao});")
                else:
                    mycursor.execute(f"INSERT INTO tb_funcionarios
(CPF_funcionario, nome, email, senha, SN, permissao, id_funcao)
VALUES    ('{cpf}','{nome}',    '{email}',    '{senha_criptografada}',
'{sn}', '{permissao}', {id_funcao});")
                myBD.commit()

                return True
            except:

                return False
```

Apêndice C – Tela Login do EASY-REQUEST

Este apêndice contém os códigos responsáveis pela implementação da Tela de Login. A Tela de Login inclui um formulário que coleta o SN e a senha do usuário, para criar uma sessão em seu navegador.

No HTML da Tela de Login existe um formulário com campos que coletam o SN e a Senha do usuário. Além do formulário existem mais dois containers para a foto de fundo e a logo do Easy Request.

```
    <!DOCTYPE html>
    <html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <!-- Link para o arquivos CSS -->
        <link rel="stylesheet" href="/static/CSS/global.css">
        <link rel="stylesheet" href="/static/CSS/RF002-log.css">
        <script defer src="/static/JS/RF002-log.js"></script>
        <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <!-- Link para a biblioteca SweetAlert-->
```

```
        <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
        <!-- link JS biblioteca scrollrevealjs -->
        <script src="https://unpkg.com/scrollreveal"></script>
        <!-- Logo do Easy Request que aparece na aba do navegador -
->
        <link       rel="shortcut       icon"       type="image/png"
href="/static/IMG/TASK request.svg">

        <title>Login</title> <!-- Título da página que aparece na
aba do navegador -->
    </head>
    <body>
        <main>
            <!-- Container para a inserção da imagem de fundo -->
            <section class="container-main">
                <!-- Esta seção pode ser usada para adicionar uma
imagem de fundo ou qualquer outro conteúdo visual -->
            </section>

            <!-- Container para o formulário de login -->
            <div class="container-form">
                <div class="form">
                    <h1>Login</h1> <!-- Título do formulário -->

                    <!-- Rótulo e campo de entrada para o SN -->
                    <label for="inp-SN">SN:</label>
                    <input placeholder="Digite seu SN" type="text"
name="inp-SN" id="inp-SN" required>

                    <!-- Rótulo e campo de entrada para a senha -->
                    <label for="inp-senha">Senha:</label>
                    <div class="senha-container">
                        <input   placeholder="Digite   sua   senha"
type="password" name="inp-senha" id="inp-senha" required>

                        <!-- Botão para mostrar/ocultar a senha -->
                        <button    type="button"    id="toggleSenha"
class="container-form__toggle-button">
                            <img                  id="cadeadoIcone"
src="/static/IMG/fechar-o-olho.png" alt="Mostrar senha" />
                        </button>
                    </div>

                    <!-- Botão para concluir o login -->
                    <button     id="btnLogin"     onclick="logar()"
class="container-form__btn-envio">Concluir Login</button>
                </div>
            </div>
```

```
        </main>

        <!-- Rodapé da página -->
        <footer>
            <section     class="container-footer"     id="container-
footer">
                <img     src="/static/IMG/Brand     Identity     and
Design.svg" alt="Logo Easy Request"> <!-- Logo da empresa no rodapé
-->
            </section>
        </footer>
    </body>
    </html>
```

Na Tela de Login, o CSS é utilizado para estilizar o formulário, posicionar bem os elementos na tela e permitir que a página se adapte a diferentes tipos de tela por meio da regra “@media”, que define algumas alterações de layout de acordo com a largura da tela.

```
    body{
        background-color: var(--color06);
        display: flex;
        align-items: center;
        justify-content: center;
        flex-direction: column;
    }

    /* Estilo da Imagem no Topo */
    .container-main{
        width: 100vw;
        height: 30vh;
        z-index: -1;
        background: url(/static/IMG/Telacad.png);
        background-repeat: no-repeat;
        background-size: cover;
        border-bottom-left-radius: 2rem;
        border-bottom-right-radius: 2rem;
    }

    /* Estilo do Forms */
    main{
        position: relative;
        display: flex;
        justify-content: center;
        align-items: center;
```

```
        flex-direction: column;
    }
    .container-form{
        position: absolute;
        top:20rem ;
        z-index: 1;
        background-color: var(--color01);
        width: 37rem;
        height: 40rem;
        border-radius: 2rem;
        position: absolute;
        box-shadow:  5px 10px 18px var(--color02);
    }

    .container-form .form{
        display: flex;
        justify-content: center;
        align-items: center;
        flex-direction: column;
        margin: 3rem;
    }

    /* Estilo do Titulo Login */
    .container-form .form h1{
        margin-bottom: 1rem;
        font-size: 3.5rem;
    }

    .container-form .form label{
        margin:1rem 23rem 1rem 0;
        font-weight: bold;
        font-size: 1.5rem;
    }

    /* Estilo dos Input (Chamado Por ID) */
    #inp-SN , #inp-senha{
        background-color: var(--color04);
        border: none;
        border-radius: 5rem;
        padding: 1rem;
        width: 100%;
    }

    #inp-SN{
        margin-bottom: 2rem;
    }

    /* Estilo do Botão "Confirmar Login" */
    button{
```

```
        display: inline-block;
        width: 30rem;
        padding: 1.2rem;
        text-align: center;
        border-radius: 50px;
        font-size: 1.6rem;
        font-weight: bold;
        text-decoration: none;
        transition: all 0.3s ease-in-out;
        cursor: pointer;
        background-color: var(--color03);
        border: 2px solid var(--color03);
        color: var(--color01);
        margin-bottom: 1rem;
        margin-top: 5rem;
    }

    /* Estilo da Logo */
    footer{
        width: 100vw;
        display: flex;
        justify-content: center;
    }
    .container-footer{
        background-color: var(--color01);
        width: 45vw;
        height: 10rem;
        position: fixed;
        bottom: 0;
        border-top-right-radius: 2rem;
        border-top-left-radius:2rem ;
        display: flex;
        justify-content: center;
        align-items: center;

    }

    .container-footer img{
        width: 14rem;
        height: auto;
    }



    /* -------------RESPONSIVIDADE-------------- */


    @media (max-width: 926px){
```

```
        button {
            width: 25rem;
            padding: 1rem;
            font-size: 1.5rem;
            margin-bottom: 1rem;
            margin-top: 1rem;
        }

        .container-form{
            top: 15rem;
            width: 32rem;
            height: 31rem;
        }


        #inp-SN {
            margin-bottom: 1rem;
        }
    }

    @media(min-width: 1024px){
        .container-form{
            top: 10rem;
            width: 35rem;
            height: 37rem;
        }

        .container-footer{
            width: 32vw;
        }
    }

    @media(min-width:1100px){
        .container-form{
            top: 15rem;
            width: 38rem;
            height: 37rem;
        }

        .container-footer{
            width: 29vw;
        }
    }

    .senha-container{
        display: flex;
        width: 100%;
        align-items: center;
        flex-direction: row;
```

```
        gap: 0.5rem;
    }
    .senha-container button{
        width: 2rem;
        height: auto;
        margin: 0;
        padding: 0.5rem 1.5rem;
        display: flex;
        align-items: center;
        justify-content: center;
        border-radius: 1.2rem;
    }
    .senha-container img{
        width: 2rem;
        height: auto;
    }
```

O JavaScript da Tela de Login tem como principal objetivo pegar as informações do formulário e enviar para o Python. Além disso, os textos dos campos do formulário são formatados automaticamente pelo JavaScript.

```
    document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal e define a configuração global
        window.sr = ScrollReveal({
            reset: true
        });

        // Aplica o efeito de revelação ao elemento com a class
'container-main'
        sr.reveal('.container-main', {
            duration: 500,
            origin: 'top',
            distance: '50px'
        });
         // Aplica o efeito de revelação ao elemento com a class
'.container-form'
         sr.reveal('.container-form', {
            duration: 500,
            origin: 'top',
            distance: '50px'
        });


    });
```

```
    // Cria a transição do formulário
    document.addEventListener("DOMContentLoaded", function() {
        const containerForm = document.querySelector('.container-
form');

        // Adiciona a classe 'show' após um pequeno atraso para
permitir a transição
        setTimeout(() => {
            containerForm.classList.add('show');
        }, 100); // Ajuste o tempo conforme necessário
    });


    // Função para o campo de SN
    // Prefixo fixo
    const prefixo = "SN";
    const snInput = document.getElementById('inp-SN');

    // Adiciona o prefixo no campo quando o usuário clica no campo
    snInput.addEventListener('focus', function(){
        if (snInput.value === '') {
            snInput.value = prefixo;
        }else{
            return;
        }
    })

    // Remove o prefixo no campo quando o usuário clica fora do
campo
    snInput.addEventListener('blur', function(){
        if (snInput.value === prefixo) {
            snInput.value = '';
        }else{
            return;
        }
    })

    // Adiciona o evento de input
    snInput.addEventListener('input', function() {
        // Remove o prefixo para evitar duplicação
        let value = snInput.value.replace(prefixo, '');
        // Remove qualquer caractere que não seja número
        value = value.replace(/\D/g, '');
        // Limita a quantidade de números a 7
        value = value.substring(0, 7);
        // Atualiza o valor do input com o prefixo "SN" e os
números digitados
        snInput.value = prefixo + value;
    });
```

```
    // Bloqueia o usuário de deletar o prefixo "SN"
    snInput.addEventListener('keydown', function(e) {
        // Previne que o usuário apague o prefixo
        if (snInput.selectionStart < prefixo.length && (e.key ===
"Backspace" || e.key === "Delete")) {
            e.preventDefault();
        }
    });

    // Pegadno os valores dos inputs
    const sn = document.getElementById('inp-SN');
    const senha = document.getElementById('inp-senha');
    var permissao = '';

    // Função que efetua o login do usuário
    function logar(){
        if (sn.value == '' || senha.value == '') {
            Swal.fire({
                icon: "error",
                title: "Oops...",
                text: "SN ou senha não podem estar vázios!",
                showConfirmButton: false,
                timer: 3500
              });
              return;
        }
        var dados = {
            sn:sn.value,
            senha:senha.value
        }

        $.ajax({
            url: '/realizar-login',
            type: 'POST',
            data: JSON.stringify(dados),
            contentType: 'application/json',
            success: function(dados_login){
                if(dados_login['permissao'] == 'administrador'){
                    window.location.href = '/tl-administrador';
                }

                else if(dados_login['permissao'] == 'manutencao'){
                    window.location.href = '/RF006';
                }

                else if(dados_login['permissao'] == 'solicitante'){
                    window.location.href = '/tl-solicitante';
                }
```

```
                console.log(dados_login["permissao"])
            },
            error: function(){
                Swal.fire({
                    icon: "error",
                    title: "Oops...",
                    text: "SN ou senha inválidos!",
                    showConfirmButton: false,
                    timer: 3500
                  });
            }
        })
    }

    // Mostrar e ocultar senha
    document.getElementById('toggleSenha').addEventListener('click'
, function () {
        const senhaInput = document.getElementById('inp-senha');
        const                    cadeadoIcone                    =
document.getElementById('cadeadoIcone');

        if (senhaInput.type === 'password') {
            senhaInput.type = 'text';
            cadeadoIcone.src = '/static/IMG/olho-de-perto.png';
            cadeadoIcone.alt = 'Ocultar senha';
        } else {
            senhaInput.type = 'password';
            cadeadoIcone.src = '/static/IMG/fechar-o-olho.png';
            cadeadoIcone.alt = 'Mostrar senha';
        }
    });

    //Executando a função de login sempre que clicar na tecla Enter
em qualquer campo selecionado ou lugar da tela
    sn.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            logar();
        }
    });

    senha.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            logar();
        }
    });

    document.addEventListener("keypress", function (event) {
        if (event.key === "Enter") {
            logar();
```

```
        }
    });
```

No BackEnd, o Python renderiza a página de login por meio da rota “/RF002”, verifica se as informações do usuário existem no Banco de Dados por meio da rota “/realizar-login”, que executa a função “logar()”.

```
    # Criando a rota para a tela de login
    @app.route("/RF002")
    def pg_login(): # Função que renderiza a tela de login
        if "usuario" in session:
            permissao = session["usuario"]["permissao"]

            if permissao == "administrador":
                return redirect("/tl-administrador")
            elif permissao == "manutencao":
                return redirect("/RF006")
            elif permissao == "solicitante":
                return redirect("/tl-solicitante")
        else:
            return render_template("RF002-log.html")
```

```
    # Criando rota para realizar o login
    @app.route("/realizar-login", methods=["POST"])
    def realizar_login(): # Função que executa o login
        usuario = Usuario()

        dados = request.get_json()
        sn = dados["sn"]
        senha = dados["senha"]

        usuario.logar(sn, senha)

        if usuario.logado:
            myBD = Connection.conectar()

            mycursor = myBD.cursor()

            try:
                mycursor.execute(f"SELECT nome FROM tb_funcoes
WHERE id_funcao = {usuario.id_funcao}")
                funcao = mycursor.fetchone()
                login = True
            except:
```

```
                login = False

            if login:
                session["usuario"]      =      {"CPF":usuario.cpf,
"nome":usuario.nome,                                        "sn":sn,
"foto":usuario.foto,"funcao":funcao[0],
"permissao":usuario.permissao}
            else:
                session["usuario"]      =      {"CPF":usuario.cpf,
"nome":usuario.nome,                                        "sn":sn,
"foto":usuario.foto,"funcao":"Administrador",
"permissao":usuario.permissao}

            return
jsonify({'permissao':session["usuario"]["permissao"]}), 200
        else:
            session.clear()
            return {'mensagem':'ERRO'}, 500
```

```
    # Função que executa o login do usuário no sistema.
        def logar(self, sn, senha):
            myBD = Connection.conectar()

            mycursor = myBD.cursor()

            self.senha = senha
            senha_criptografada                                    =
sha256(self.senha.encode()).hexdigest()

            mycursor.execute(f"SELECT  CPF_funcionario,  nome,  SN,
foto,  permissao,  id_funcao  FROM  tb_funcionarios  WHERE  SN  =  '{sn}'
AND BINARY senha = '{senha_criptografada}';")

            resultado = mycursor.fetchone()
            print(resultado)

            if  resultado != None:
                self.logado = True
                self.cpf = resultado[0]
                self.nome = resultado[1]
                self.sn = resultado[2]
                self.foto = resultado[3]
                self.permissao = resultado[4]
                self.id_funcao = resultado[5]

            else:
                self.logado = False
```

Apêndice D – Tela Inicial do Administrador

Este apêndice contém os códigos responsáveis pela implementação da Tela Inicial do Administrador. A Tela Inicial do Administrador inclui uma barra de navegação com botões que direcionam o usuário para as telas de solicitações, relatórios e finalização, além de um local com as informações básicas do usuário. Também existem cards que contém as informações de todos os serviços encaminhados para os funcionários da manutenção. Os cards são adicionados pelo frontend(JavaScript) utilizando dados extraídos do backend(Python).

Na Tela do Administrador o código HTML contém um cabeçalho com uma barra de navegação com botões que redirecionam o usuário para as telas de solicitações, relatórios e finalização, além de um local com as informações básicas do usuário como nome e função. Também contém no em sua parte principal um container vazio, que receberá informações do frontend e um outro container que engloba um botão que redireciona para tela de solicitação.

```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta   name="viewport"   content="width=device-width,   initial-
scale=1.0">

        <title>Tela Inicial</title>

        <!-- Link Arquivo CSS Global -->
        <link rel="stylesheet" href="/static/CSS/global.css">
        <!-- Link Arquivo CSS -->
        <link rel="stylesheet" href="/static/CSS/TLadministrador.css">
        <!-- Link Arquivo JS -->
        <script defer src="/static/JS/TLadministrador.js"></script>
        <!-- Link para os ICONS -->
        <script          src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
        <!-- link Arquivo JS data -->
        <script defer src = "/static/JS/funcao-horas.js"></script>
        <script defer src="/static/JS/tamanhoNome.js"></script>
        <!-- Link para o usúario receber notificações -->
```

```
            <script                defer                type="module"
src="/static/JS/notificacoes.js"></script>

            <link                                    rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outl
ined:opsz,wght,FILL,GRAD@24,400,0,0&icon_names=add" />

            <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
            <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
            <!-- link JS biblioteca SweetAlert -->
            <script src="https://unpkg.com/scrollreveal"></script>
            <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
            <!-- Link de Estilo de Fonte -->
            <link                                    rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outl
ined:opsz,wght,FILL,GRAD@20..48,100..700,0..1,-
50..200&icon_names=add" />
            <!-- Logo do Easy Request na guia-->
            <link     rel="shortcut     icon"     type="imagex/png"
href="/static/IMG/TASK request.svg">
        </head>
        <body>
            <!-- Cabeçalho da página com barra de navegação -->
            <header>
                <section class="container-header-user">
                    <div class="info-usuario">
                        <figure><img    src="{{    campo_foto    if
campo_foto                                                    else
'https://st4.depositphotos.com/11574170/25191/v/450/depositphotos_25
1916955-stock-illustration-user-glyph-color-icon.jpg'  }}"  alt="Foto
de Perfil"></figure>
                        <div      class="container-header-user_info
horas">
                            <h2 class="container-header-user_nome">
<span class="saudacao" id="saudacao"> </span>{{campo_nome}}</h2>
                            <p             class="container-header-
user_cargo">{{campo_funcao}}</p>
                        </div>
                    </div>
                    <nav class="nav">
                        <input type="checkbox" id="menu-toggle">
                        <label for="menu-toggle" class="menu-icon">
                            <span></span>
                            <span></span>
                            <span></span>
```

```
                        </label>
                        <ul class="menu">
                            <li><a                href="/RF004"><img
src="/static/IMG/12.png" alt="Solicitações"> Solicitações</a></li>
                            <li><a        href="/RF010">        <img
src="/static/IMG/13.png" alt="Relatórios"> Relatórios</a></li>
                            <li><a        href="/RF008">        <img
src="/static/IMG/15.png" alt="Finalização"> Finalização</a></li>
                        </ul>

                        <button    class="button    button--logout"
onclick="logout()">

                            <div class="button__icon">

                              <svg viewBox="0 0 512 512">
                                <path     d="M377.9     105.9L500.7
228.7c7.2 7.2 11.3 17.1 11.3 27.3s-4.1 20.1-11.3 27.3L377.9 406.1c-
6.4 6.4-15 9.9-24 9.9c-18.7 0-33.9-15.2-33.9-33.9l0-62.1-128 0c-17.7
0-32-14.3-32-32l0-64c0-17.7 14.3-32 32-32l128 0 0-62.1c0-18.7 15.2-
33.9 33.9-33.9c9 0 17.6 3.6 24 9.9zM160 96L96 96c-17.7 0-32 14.3-32
32l0 256c0 17.7 14.3 32 32 32l64 0c17.7 0 32 14.3 32 32s-14.3 32-32
32l-64 0c-53 0-96-43-96-96L0 128C0 75 43 32 96 32l64 0c17.7 0 32
14.3 32 32s-14.3 32-32 32z"></path>
                              </svg>

                            </div>

                            <div class="button__text">Sair</div>

                          </button>

                    </nav>

                </section>
            </header>

            <!-- Conteúdo principal da página -->
            <main class="container-baixo">
                <section       class="container_baixo-ladoEsquerdo"
id="container_baixo-ladoEsquerdo">

                </section>
            </main>

            <!-- Botão que leva para a tela de solicitação -->
            <section class="contrainer-btn">
                <button                                  class="Btn"
onclick="redirectSolicitacao()">
```

```
                <div class="sign">
                  <span class="material-symbols-outlined">
                      add
                      </span>

                  <div class="text">Solicitar</div>
                </button>

            </section>
        </body>
        <script src="/static/JS/TLadministrador.js"></script>
    </html>
```

Na página inicial do administrador, o CSS é utilizado para estilizar os três componentes principais da página, que são o cabeçalho, o container que abrange os encaminhamentos e o container que abrange o botão de solicitação, tornando assim a página mais atraente e intuitiva. Ele também é responsável por permitir que a página se adapte a diferentes tipos de tela por meio da regra “@media”, que define algumas alterações de layout de acordo com a largura da tela.

```
    body{
        overflow-x:hidden;
    }
    header {
        width: 100vw;
        height: 18vh;
        display: flex;
        align-items: flex-start;
        align-items: center;
        padding-top: 2rem;
    }
    .container-header-user_info{
      margin-left: 1rem;
    }
    /* Estilo do cabeçalho */
    .container-cumprimentacao {
        width: 100vw;
        display: flex;
        justify-content: space-around;
        align-items: center;
    }
```

```
#cumprimentacao{
    align-self: center;
    font-size: 2.5rem;
    font-weight: lighter;
}
.info-usuario{
  display: flex;
}
/* Estilos gerais do menu */
.nav {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.nav ul {
  list-style: none;
  display: flex;
  gap: 2rem;
  padding-right: 1.6rem;
}

.nav ul li a {
  position: relative;
  text-decoration: none;
  color: var(--color05);
  font-size: 1.5rem;
  font-weight: 600;
  padding-bottom: 5px;
}

.nav ul li a::after {
  content: '';
  position: absolute;
  left: 50%;
  bottom: 0;
  width: 0;
  height: 2px;
  background-color: var(--color05);
  transition: width 0.4s ease, left 0.4s ease;
}

.nav ul li a:hover::after {
  width: 100%;
  left: 0;
}
.menu-icon {
  display: none;
```

```
    cursor: pointer;
    flex-direction: column;
    gap: 0.4rem;
  }

  .menu-icon span {
    width: 25px;
    height: 3px;
    background-color: var(--color06);
  }

  #menu-toggle {
    display: none;
  }
  .img-menu img{
    width: auto;
    height: 2.5rem;
  }
  #menu-toggle:checked + .menu-icon + .menu {
    display: flex;
    position: absolute; /* ou position: absolute; */
    z-index: 10000;
    margin-top: 2.5rem;
  }
  .button {
    display: flex;
    align-items: center;
    justify-content: flex-start;
    width: 4rem; /* Largura inicial do botão */
    height: 4rem;
    border: none;
    border-radius: 50%;
    margin: auto;
    cursor: pointer;
    position: relative;
    overflow: hidden;
    transition-duration: .3s;
    box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.199);
    background-color: var(--color05);
    margin-right: 1.5rem;
  }

  .button:hover {
    width: 10rem; /* Limita o tamanho no hover para 10rem */
    border-radius: 40px;
    transition-duration: .3s;
  }

  .button__icon {
```

```
    width: 100%;
    transition-duration: .3s;
    display: flex;
    align-items: center;
    justify-content: center;
  }

  .button__icon svg {
    width: 1.5rem;
  }

  .button__icon svg path {
    fill: white;
  }

  .button__text {
    position: absolute;
    right: 0%;
    width: 0%;
    opacity: 0;
    color: white;
    font-size: 1.2rem;
    font-weight: 600;
    transition-duration: .3s;
  }

  .button:hover .button__icon {
    width: 26%;
    transition-duration: .3s;
    padding-left: 10px;
  }

  .button:hover .button__text {
    opacity: 1;
    width: 60%; /* Reduz a expansão do texto */
    transition-duration: .3s;
    padding-right: 1rem; /* Ajusta o espaçamento do texto */
  }

  .button:active {
    transform: translate(2px, 2px);
  }

  /* Estilo da Área Usuario */
  .container-header-user{
      width: 100%;
      display: flex;
      align-items: center;
      padding: 0rem 0rem 0rem 5rem;
```

```
        justify-content: space-between;
    }

    .container-header-user img{
        width: 6rem;
        height: 6rem;
        object-fit: cover;
        border-radius: 100%;
    }

    .container-header-user_nome{
        font-size: 2rem;
        color: var(--color06);
    }

    .container-header-user_cargo{
        background-color: var(--color05);
        color: var(--color01);
        min-width: 1rem;
        width: 20rem;
        margin-right: 1rem;
        text-align: center;
        padding: 0.4rem;
        border-radius: 0.5rem;
        font-size: 1.3rem;
        margin-top: 0.5rem;
    }
    /* Estilo do css do main do site */
    .container-baixo{
        width: 100vw;
        min-height: 68vh;
        max-height: 68vh;
        background-color: var(--color06);
        margin-bottom: 0;
        position: absolute;
        bottom: 0;
        left: 0;
        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        padding: 0 1.8rem;
        box-shadow: 0 -4px 10px rgba(20, 23, 53, 0.3);
        overflow-y: scroll;

    }
    /* barra de rolagem transparente */
    ::-webkit-scrollbar {
      background-color: transparent;
    }
```

```
.container_baixo-ladoEsquerdo,.container_baixo-ladoDireito{
    min-height: 50%;
    margin-top: 3rem;
    padding: 1rem;
}

.container_baixo-ladoEsquerdo h3{
    text-align: center;
    color: white;
    grid-column: span 3;
}

h2{
    font-size: 1.8rem;
    font-weight: bold;
    color: var(--color01);
}
hr{
    margin-bottom: 2rem;
}
.destaque-txt{
    color: var(--color03);
}

.ladoEsquerdo-card, .ladoDireito-card{
    background: var(--color04);
    height: 8rem;
    border-radius: 10px;
    display: flex;
    flex-direction: column;
    justify-content: center;
    font-size: 1.8rem;
    gap: 0.5rem;
    margin-top: 2rem;
    text-transform: capitalize;
    transition-duration: 1s;
}

.ladoEsquerdo-conteudo{
  cursor: pointer;
}

/* Cores na mudança do card de acordo com a urgencia */
.red:hover{
  color: var(--color04);
  background-color: var(--colorRed);
}

.green:hover{
```

```
    background-color: var(--colorGreen);
    color:var(--color04);
  }

  .yellow:hover{
    background-color: var(--color03);
    color:var(--color04);
  }

  .card-conteudo{
      display: flex;
      justify-content: space-around;
      align-items: center;
      font-weight: 700;
      transition: transform 0.2s;
  }

  .card-conteudo i:hover{
    transform: scale(1.1);
    color: var(--color04);
    border: var(--color04);
  }

  .horario{
      padding-left: 1.4rem;
  }
  .urgencia-ind {
      padding: 0.1rem 0.5rem;
      border: 0.2rem solid var(--color02);
      border-radius: 0.5rem;
      color: var(--color02);
      cursor: default;
  }


  /*Cores de mudança do indicador de urgencia  */
  .urgencia-yellow{
      border: 0.2rem solid var(--color04);
      border-radius: 0.5rem;
      color: var(--color01);
      background-color:var(--color03) ;
  }

  .urgencia-red{
    border: 0.2rem solid var(--color04);
    border-radius: 0.5rem;
    color: var(--color01);
    background-color:var(--colorRed) ;
  }
```

```
      .urgencia-green{
        border: 0.2rem solid var(--color04);
        border-radius: 0.5rem;
        color: var(--color01);
        background-color:var(--colorGreen) ;
      }

      /* Estilo da Seta do Card */
      .fa-arrow-right:before {
          content: "\f061";
          color: var(--color06);
          padding: 0.5rem;
          border: 0.2rem solid var(--color06);
          border-radius: 10rem;
      }

      .botao-chamado {
          position: fixed;
          right: 20px;
          bottom: 20px;
          background-color: var(--color03);
          color: #000;
          font-size: 1.5rem;
          font-weight: bold;
          padding: 0.7rem 1rem; /* Padding inicial */
          border-radius: 2rem;
          box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);
          display: flex;
          align-items: center;
          justify-content: center;
          cursor: pointer;
          text-decoration: none;
          z-index: 1000;
          transition: background-color 0.3s ease, transform 0.3s
ease; /* Transições suaves */
      }
      .contrainer-btn{
      position: absolute;
      }
      /* From Uiverse.io by vinodjangid07 */
      .Btn {
          display: flex;
          align-items: center;
          justify-content: flex-start;
          width: 45px;
          height: 45px;
          border: none;
          border-radius: 50%;
```

```
          cursor: pointer;
          position: fixed; /* Mudei para fixed */
          bottom: 20px; /* Distância do fundo */
          right: 20px; /* Distância da direita */
          overflow: hidden;
          transition-duration: .3s;
          box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.199);
          background-color:var(--color03);
          color: #000;
          z-index: 1000; /* Garante que o botão fique sobre outros
elementos */
      }


        /* plus sign */
        .sign {
          width: 100%;
          transition-duration: .3s;
          display: flex;
          align-items: center;
          justify-content: center;
        }

        .sign svg {
          width: 17px;
        }

        .sign svg path {
          fill: rgb(0, 0, 0);
        }
        /* text */
        .text {
          position: absolute;
          right: 0%;
          width: 0%;
          opacity: 0;
          color: rgb(0, 0, 0);
          font-size: 1.2em;
          font-weight: 600;
          transition-duration: .3s;
        }
        /* hover effect on button width */
        .Btn:hover {
          width: 125px;
          border-radius: 40px;
          transition-duration: .3s;
        }

        .Btn:hover .sign {
```

```
      width: 30%;
      transition-duration: .3s;
      padding-left: 20px;
    }
    /* hover effect button's text */
    .Btn:hover .text {
      opacity: 1;
      width: 70%;
      transition-duration: .3s;
      padding-right: 10px;
    }
    /* button click effect*/
    .Btn:active {
      transform: translate(2px ,2px);
    }

  /* -------------MEDIA---------------- */
  @media (max-width: 926px){
    .container-header-user{
        width: 100vw;
        display: flex;
        align-items: center;
        padding: 0rem 0rem 0rem 1rem;
        justify-content: space-between;
    }
    .container-baixo{
      width: 100vw;
      min-height: 74vh;

    }

  }
  /* Para dispositivos menores, como tablets */
  @media (max-width: 768px) {
    .nav ul {
        display: none;
        flex-direction: column;
        position: absolute;
        top: 70px;
        right: 0;
        width: 200px;
        background-color: var(--color04);
        padding: 1rem;
        border-radius: 0.5rem;
        box-shadow: 0 5px 10px rgba(0, 0, 0, 0.3);
    }

    .menu-icon {
        display: flex;
```

```
            margin-right: 2rem;
        }

        /* Quando o checkbox é clicado, mostra o menu */
        #menu-toggle:checked + .menu-icon + .menu {
            display: flex;
        }

        .container-baixo{
          width: 100vw;
          min-height: 74vh;

        }

    }

    /* Para dispositivos como celulares */
    @media (max-width: 400px){
        .container-header-user{
          padding: 0;
        }

        .container-header-user_cargo{
          width: 15rem;
        }
        .container-header-user_nome{
          font-size: 1.5rem;
        }

    }

    .menu img{
      width: 2.5rem;
      height: auto;
      border-radius: 0;
    }
```

O JavaScript da tela inicial do administrador tem como principal objetivo criar funcionalidades para os botões por meio das funções “redirectSolicitacao()”, “logout()” e “redirectDetalhes()” e adicionar os encaminhamentos na tela a cada 5 segundos e criar uma animação para quando a tela for carregada.

```
    document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal e define a configuração global
        window.sr = ScrollReveal({
```

```
            reset: true  // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Aplica o efeito de revelação ao elemento com o ID 'info-
usuario'
        sr.reveal('.info-usuario', {
            duration: 500,  // Duração da animação em milissegundos
            origin: 'top',   // Elemento surge do topo
            distance: '50px' // Distância percorrida pelo elemento
        });

        // Aplica o efeito de revelação ao elemento com o ID
'conteudo-principal'
        sr.reveal('.container-baixo', {
            duration: 800,  // Duração da animação (mais lenta)
            origin: 'bottom',  // Elemento surge de baixo
            distance: '100px',  // Distância que o elemento vai
percorrer
            delay: 200,  // Atraso antes de iniciar a animação
            easing: 'ease-in-out',   // Tipo de suavização do
movimento
        });
    });

    // Função para ir para a tela de fazer solicitações
    function redirectSolicitacao() {
        window.location.href = '/RF003';
    }

    // Função para deslogar da conta
    function logout() {
        window.location.href = '/logout';
    }

    // Função que mostra todos os encaminhamentos que ainda não
foram finalizados
    function retorna_encaminhados(){
        $.ajax({
            url: '/retorna-encaminhados',
            type: 'GET',
            success: function(encaminhado){
                document.getElementById('container_baixo-
ladoEsquerdo').innerHTML = `
                    <div class="ladoEsquerdo-txt">
                        <h2>Serviços já<span class="destaque-txt">
Encaminhados</span></h2>
                        <hr>
                    </div>`;
```

```
                if (encaminhado.length == 0) {
                    document.getElementById('container_baixo-
ladoEsquerdo').innerHTML  +=  '<h3>Não  existem  encaminhamentos  em
aberto</h3>';
                } else {
                    console.log(encaminhado)
                    for (let x = 0; x < encaminhado.length; x++) {
                        var infoEncaminhadosYellow =`
                        <div          class="ladoEsquerdo-conteudo"
onclick=redirectDetalhes(${encaminhado[x]['id_solicitacao']},${encam
inhado[x]['id_encaminhamento']})>
                            <div class="ladoEsquerdo-card yellow">
                                <p               class="horario">|
${encaminhado[x]['status']}</p>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhado[x]['id_sala']}|${encaminhado[x]['bloco']
}</p>
                                    <p          class="urgencia-ind
urgencia-yellow" id="urgencia-ind">${encaminhado[x]['urgencia']}</p>
                                    <a             href="/detalhes-
encaminhamento/${encaminhado[x]['id_solicitacao']}/${encaminhado[x][
'id_encaminhamento']}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                            </div>
                        </div>`;

                        var infoEncaminhadosRed =`
                        <div          class="ladoEsquerdo-conteudo"
onclick=redirectDetalhes(${encaminhado[x]['id_solicitacao']},${encam
inhado[x]['id_encaminhamento']})>
                            <div class="ladoEsquerdo-card red">
                                <p               class="horario">|
${encaminhado[x]['status']}</p>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhado[x]['id_sala']}|${encaminhado[x]['bloco']
}</p>
                                    <p          class="urgencia-ind
urgencia-red" id="urgencia-ind">${encaminhado[x]['urgencia']}</p>
                                    <a             href="/detalhes-
encaminhamento/${encaminhado[x]['id_solicitacao']}/${encaminhado[x][
'id_encaminhamento']}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                            </div>
                        </div>`;

                        var infoEncaminhadosGreen =`
```

```
                        <div          class="ladoEsquerdo-conteudo"
onclick=redirectDetalhes(${encaminhado[x]['id_solicitacao']},${encam
inhado[x]['id_encaminhamento']})>
                            <div class="ladoEsquerdo-card green">
                                <p                class="horario">|
${encaminhado[x]['status']}</p>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhado[x]['id_sala']}|${encaminhado[x]['bloco']
}</p>
                                    <p          class="urgencia-ind
urgencia-green" id="urgencia-ind">${encaminhado[x]['urgencia']}</p>
                                    <a             href="/detalhes-
encaminhamento/${encaminhado[x]['id_solicitacao']}/${encaminhado[x][
'id_encaminhamento']}"><i class="fa-solid fa-arrow-right "></i></a>
                                </div>
                            </div>
                        </div>`;

                        // Atualiza o status do card de acordo com
o status do item
                        if (encaminhado[x]['urgencia'] === 'media')
{
                            document.getElementById('container_baix
o-ladoEsquerdo').innerHTML += infoEncaminhadosYellow;
                        } else if (encaminhado[x]['urgencia'] ===
'alta') {
                            document.getElementById('container_baix
o-ladoEsquerdo').innerHTML += infoEncaminhadosRed;
                        } else if (encaminhado[x]['urgencia'] ===
'baixa') {
                            document.getElementById('container_baix
o-ladoEsquerdo').innerHTML += infoEncaminhadosGreen;
                        };


                    }
                }
            }
        })
    }

    //  Executa  a  função  retorna_encaminhamentos  assim  que  o
documento é carregado
    $(document).ready(retorna_encaminhados)

    // Executa a função retorna_encaminhamentos a cada 5 segundos
    setInterval(() => {
        retorna_encaminhados()
```

```
        console.log("Recarregado!")
    }, 5000);

    // Função para ir para a tela de detalhes do encaminhamento
    function redirectDetalhes(id_solicitacao, id_encaminhamento) {
        window.location.href = `/detalhes-encaminhamento/${id_solicitacao}/${id_encaminhamento}`;
    }
```

No BackEnd, o Python(por meio do Flask) renderiza a página e adiciona o nome e função do usuário no HTML utilizando a rota “/tl-administador”. Já pela rota chamada “/retorna-encaminhados”, o Python executa a função “mostrar_encaminhados()” com o objetivo de coletar no Banco de Dados as informações relacionadas aos encaminhamentos, para posteriormente essas informações serem adicionadas ao HTML por meio do JavaScript.

```
    # Criando a rota para a tela do administrador
    @app.route("/tl-administrador")
    def tl_administrador(): # Função que renderiza a tela do administrador
        if "usuario" in session:
            funcao = session["usuario"]["funcao"]
            nome_completo = session["usuario"]["nome"]

            # Dividindo o nome em partes
            partes_nome = nome_completo.split()

            # Verificando se o nome tem mais de uma parte
            if len(partes_nome) > 1:
                primeiro_ultimo_nome = f"{partes_nome[0]} {partes_nome[-1]}"
            else:
                # Caso o nome tenha apenas uma parte, retorna somente essa parte
                primeiro_ultimo_nome = partes_nome[0]

            return render_template("TLadministrador.html", campo_nome = primeiro_ultimo_nome, campo_funcao = funcao)
        else:
            return redirect("/")
```

```
    # Criando a rota para retornar os serviços que foram encaminhados para algum funcionário
```

```
    @app.route("/retorna-encaminhados")
    def retorna_encaminhados(): # Função que retorna os serviços que foram encaminhados para algum funcionário
        encaminhamento = Encaminhamento()
        encaminhado = encaminhamento.mostrar_encaminhados()
        return jsonify(encaminhado), 200
```

```
        # Mostra todos os encaminhamentos que ainda não foram finalizados
        def mostrar_encaminhados(self):
            myBD = Connection.conectar()

            mycursor = myBD.cursor()

            mycursor.execute(f"SELECT s.id_sala, s.bloco, enc.urgencia, enc.status, sol.id_solicitacao, enc.id_encaminhamento FROM tb_funcionarios func , tb_encaminhamentos enc, tb_solicitacoes sol, tb_salas s WHERE enc.id_solicitacao = sol.id_solicitacao AND sol.id_sala = s.id_sala AND enc.CPF_funcionario = func.CPF_funcionario AND enc.status_finalizacao = FALSE AND (enc.status = 'fazendo' OR enc.status = 'à fazer') ORDER BY id_encaminhamento DESC;")

            mostrar_encaminhados= mycursor.fetchall()

            lista_encaminhado = []
            for encaminhado in mostrar_encaminhados:
                lista_encaminhado.append({
                    "id_sala":encaminhado[0],
                    "bloco":encaminhado[1],
                    "urgencia":encaminhado[2],
                    "status":encaminhado[3],
                    "id_solicitacao":encaminhado[4],
                    "id_encaminhamento":encaminhado[5]
                })

            return lista_encaminhado
```

Apêndice E – Tela Inicial da Manutenção - EASY-REQUEST

Neste apêndice, é apresentado o Desing da Tela Inicial da Manutenção feita a partir do código HTML, a qual foi personalizada com elementos a partir da estilização em CSS. Aonde apresenta a estrutura da tela Inicial do Técnico de Manutenção no sistema. Ele exibe informações como o perfil do usuário logado, serviços em aberto e serviços encaminhados.

```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta    name="viewport"    content="width=device-width,
initial-scale=1.0">

        <title>Tela Inicial</title>

         <!-- Link Arquivo CSS Global -->
         <link rel="stylesheet" href="/static/CSS/global.css">
         <!-- Link Arquivo CSS -->
         <link     rel="stylesheet"     href="/static/CSS/RF006-
TLmanuten.css">
         <!-- Link Arquivo JS -->
         <script          defer          src="/static/JS/RF006-
TLmanuten.js"></script>
         <!-- Link para os ICONS -->
        <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
        <!-- link Arquivo JS data -->
         <script         defer         src="/static/JS/funcao-
data.js"></script>
         <script     defer     src     =     "/static/JS/funcao-
horas.js"></script>
         <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <!-- Importando fontes do google -->
        <link                                    rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outl
ined:opsz,wght,FILL,GRAD@24,400,0,0&icon_names=add" />
        <!-- link JS biblioteca SweetAlert -->
        <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
        <script src="https://unpkg.com/scrollreveal"></script>
```

```
            <!-- Link para o usúario receber notificações -->
            <script              defer              type="module"
src="/static/JS/notificacoes.js"></script>

            <!-- Logo do Easy Request na guia-->
            <link     rel="shortcut     icon"     type="imagex/png"
href="/static/IMG/TASK request.svg">
        </head>
        <body>
            <header>
                <section class="container-header-user">
                    <div class="info-usuario">
                        <figure><img    src="{{    campo_foto    if
campo_foto                                                    else
'https://st4.depositphotos.com/11574170/25191/v/450/depositphotos_25
1916955-stock-illustration-user-glyph-color-icon.jpg'  }}"  alt="Foto
de Perfil"></figure>
                        <div      class="container-header-user_info
horas">
                            <h2 class="container-header-user_nome">
<span class="saudacao" id="saudacao"> </span>{{campo_nome}}</h2>
                            <p             class="container-header-
user_cargo">{{campo_funcao}}</p>
                        </div>
                    </div>
                    <!-- Botão para deslogar (sair da conta) -->
                    <button      class="button      button--logout"
onclick="logout()">

                        <div class="button__icon">

                          <svg viewBox="0 0 512 512">
                            <path  d="M377.9  105.9L500.7  228.7c7.2
7.2 11.3 17.1 11.3 27.3s-4.1 20.1-11.3 27.3L377.9 406.1c-6.4 6.4-15
9.9-24 9.9c-18.7 0-33.9-15.2-33.9-33.9l0-62.1-128 0c-17.7 0-32-14.3-
32-32l0-64c0-17.7 14.3-32 32-32l128 0 0-62.1c0-18.7 15.2-33.9 33.9-
33.9c9 0 17.6 3.6 24 9.9zM160 96L96 96c-17.7 0-32 14.3-32 32l0 256c0
17.7 14.3 32 32 32l64 0c17.7 0 32 14.3 32 32s-14.3 32-32 32l-64 0c-
53 0-96-43-96-96L0 128C0 75 43 32 96 32l64 0c17.7 0 32 14.3 32 32s-
14.3 32-32 32z"></path>
                          </svg>

                        </div>

                        <div class="button__text">Sair</div>

                      </button>
                </section>
            </header>
```

```
            <main class="container-baixo">
                <section class="container_baixo-ladoEsquerdo">
                    <div class="ladoEsquerdo-txt">
                        <h2>Serviço   em   <span   class="destaque-
txt">Aberto</span></h2>
                        <hr>
                    </div>
                    <div             class="ladoEsquerdo-conteudo"
id="encaminhamentos_fazendo">
                    </div>
                </section>
                <!-- Serviços encaminhados -->
                <div                    class="container-servico-
encaminhado_subtitulo">
                    <h2>Serviços Encaminhados</h2>
                    <hr>
                </div>
                <section class="container-servico-encaminhado">
                    <div          class="container-encaminhamento"
id="encaminhamentos_pendentes">
                    </div>

            </main>
            <button onclick="redirectSolicitacao()" class="Btn" >

                <div class="sign">
                  <span class="material-symbols-outlined">
                      add
                  </span>

                <div class="text">Solicitar</div>
            </button>
        </body>
    </html>
```

Neste CSS aplicado à tela inicial do Técnico de Manutenção organiza os elementos, com destaque para o cabeçalho, onde são exibidos o cumprimento personalizado e as informações do usuário. A navegação e interação são facilitadas através de animações suaves nos botões e cards, que indicam a urgência dos serviços encaminhados. O layout é responsivo, adaptando-se a diferentes tamanhos de tela.

```
/* Barra de rolagem escondida */
body{
  overflow-x: hidden;
}
/* Tamanho do cabeçalho */
header{
  width: 100vw;
  height: 18vh;
  display: flex;
}

/* Estilo do cabeçalho */
.container-cumprimentacao {
  width: 100vw;
  display: flex;
  justify-content: space-around;
  align-items: center;
}

#cumprimentacao{
  align-self: center;
  font-size: 2.5rem;
  font-weight: lighter;
}
.info-usuario{
  display: flex;
}
/* Estilo da data */
.container-header-data{
  margin: 1rem;
  display: flex;
  gap: 0.5rem;
  justify-content: center;
  align-items: center;
  font-size: 1.5rem;
}
.data{
  display: flex;
  gap: 0.5rem;
  font-size: 1.8rem;
}
.container-header-user_info {
margin-left: 1rem;
}
.dia-semana{
  color: var(--color05);
  font-weight: 600;
}
```

```css
/* Estilo da Área Usuario */
.container-header-user{
  width: 100vw;
  display: flex;
  align-items: center;
  padding: 0rem 0rem 0rem 2rem;
  justify-content: space-between;
}

.container-header-user img{
  width: 6rem;
  height: 6rem;
  object-fit: cover;
  border-radius: 100%;
}
.container-header-user_info{
margin-left: 1rem;
}
.container-header-user_nome{
  font-size: 1.6rem;
  color: var(--color06);
}

.container-header-user_cargo{
background-color: var(--color05);
padding: 0.4rem;
max-width: 20rem;
font-size: 1.2rem;
text-align: center;
border-radius: 0.5rem;
color: var(--color01);
margin-top: 0.2rem;
word-wrap: break-word;
}

.button {
  display: flex;
  align-items: center;
  justify-content: flex-start;
  width: 4rem; /* Largura inicial do botão */
  height: 4rem;
  border: none;
  border-radius: 50%;
  margin: auto;
  cursor: pointer;
  position: relative;
  overflow: hidden;
  transition-duration: .3s;
  box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.199);
```

```
    background-color: var(--color05);
    margin-right: 1.5rem;
  }

  .button:hover {
    width: 10rem; /* Limita o tamanho no hover para 10rem */
    border-radius: 40px;
    transition-duration: .3s;
  }

  .button__icon {
    width: 100%;
    transition-duration: .3s;
    display: flex;
    align-items: center;
    justify-content: center;
  }

  .button__icon svg {
    width: 1.5rem;
  }

  .button__icon svg path {
    fill: white;
  }

  .button__text {
    position: absolute;
    right: 0%;
    width: 0%;
    opacity: 0;
    color: white;
    font-size: 1.2rem;
    font-weight: 600;
    transition-duration: .3s;
  }

  .button:hover .button__icon {
    width: 26%;
    transition-duration: .3s;
    padding-left: 10px;
  }

  .button:hover .button__text {
    opacity: 1;
    width: 60%; /* Reduz a expansão do texto */
    transition-duration: .3s;
    padding-right: 1rem; /* Ajusta o espaçamento do texto */
  }
```

```
.button:active {
  transform: translate(2px, 2px);
}
/* Estilo serviços encaminhados */

.container-servico-encaminhado{
  width: 100%;
  height: 45vh;
  margin:1rem 1.5rem 0rem 1rem;
  overflow-y: scroll;
}

/* estilo da barra de rolagem */
::-webkit-scrollbar {
background-color: transparent;
}

.container-servico-encaminhado h2{
  font-size: 2rem;
  color: var(--color01);
  margin-left: 1rem;
}

/* Para telas normais (padrão) */
.container-encaminhamento {
  display: grid;
  text-align: center;
  height: auto;
  grid-template-columns: 1fr; /* Uma coluna */
  gap: 1rem; /* Espaço entre os itens */
  width: 100%;
  justify-items: center;

}

.container-encaminhamento h3{
color: white;
text-align: center;
grid-column: span 3;
}

.container-encaminhamento a{
  text-decoration: none;
  color: var(--color02);
  width: 100%;
  margin: 0rem 1rem 0rem 1rem;

}
```

```
.servico-encaminhado{
  background-color: var(--color01);
  height: 100%;
  width: 95%;
  margin: 1rem 0rem 7rem 0rem;
  border-radius: 1rem;
  position: relative;
  display: flex;
  flex-direction:column ;
  justify-content: center;
  align-items: center;
  text-decoration: none;
  border: solid 3px transparent;
  transition: 0.3s ease-in-out;
  gap: 0.5rem;

}

.servico-encaminhado:hover{
border: solid var(--color03) 3px;
}

.servico-encaminhado-local{
  text-align: center;
  font-size: 1.5rem;
  font-weight: bold;

}

.servico-encaminhado-data{
  background-color: var(--color04);
  position: absolute;
  top: 0;
  left: 0;
  z-index: 1;
  padding: 1rem 3rem;
  border-top-left-radius: 0.5rem;
  border-bottom-right-radius: 2rem;
  display: flex;
  gap: 0.5rem;
  font-size: 1.5rem ;
  box-shadow: -1px 2px 8px var(--color02);

}

.servico-encaminhado-dia-semana{
  font-weight: bold;
```

```
}

.urgencia-encaminhado{
  margin: 1rem;
 font-size: 1.5rem;
 padding: 0.5rem;
  border: 0.2rem solid var(--colorRed);
  border-radius: 3rem;
}

.urgencia,.urgencia-encaminhado{
  text-transform: capitalize;
}



/* Estilo do css do man do site */
.container-baixo{
  width: 100vw;
  min-height: 72vh;
  max-height: 72vh;
  overflow-y: auto; /* Adiciona rolagem vertical */
  overflow-x: hidden; /* Esconde a rolagem horizontal */
  height: auto;
  background-color: var(--color06);
  margin-bottom: 0;
  position: absolute;
  bottom: 0;
  left: 0;
  border-top-left-radius: 2rem;
  border-top-right-radius: 2rem;
  padding: 1.8rem;
  box-shadow: 0 -4px 10px rgba(20, 23, 53, 0.3);
}
.container_baixo-ladoEsquerdo,.container_baixo-ladoDireito{
  min-height: 50%;
  margin-top: 1rem;
  padding: 1rem;
}
h2{
  font-size: 1.8rem;
  font-weight: bold;
  color: var(--color01);
}

.ladoEsquerdo-conteudo{
color: white;
text-align: center;
grid-column: span 3;
```

```
}

hr{
  margin-bottom: 2rem;
}
.destaque-txt{
  color: var(--color03);
}
.ladoEsquerdo-card, .ladoDireito-card{
  background: var(--color04);
  height: 8rem;
  border-radius: 10px;
  display: flex;
  flex-direction: column;
  justify-content: center;
  font-size: 1.8rem;
  gap: 0.5rem;
  cursor: pointer;
  border: 3px solid transparent;
  transition-duration: 0.2s;
}

.ladoEsquerdo-card:hover{
  border: 3px solid var(--color03);
}

.card-conteudo{
  display: flex;
  justify-content: space-around;
  align-items: center;
  font-weight: 700;
}
.horario{
  display: none;
  padding-left: 1.4rem;
}

.bloco{
color: var(--color02);
}

.urgencia{
  color: var(--color01);
  background-color: var(--color05);
  padding: 0.6rem;
  border-radius: 0.5rem;
}

/* Cores da urgencia*/
```

```
.yellow{
  background-color: var(--color03);
}

.red{
  background-color: var(--colorRed);
}

.green{
  background-color: var(--colorGreen);
}

/* Cores da urgencia Pendente */
/* Cores na mudança do card de acordo com a urgencia */
.red_pendente{
  color: var(--color04);
  background-color: var(--colorRed);
  border: none;
  font-weight: 700;
}

.green_pendente{
  background-color: var(--colorGreen);
  color:var(--color04);
  border: none;
  font-weight: 700;
}

.yellow_pendente{
  background-color: var(--color03);
  color:var(--color04);
  border: none;
  font-weight: 700;
}

.fa-arrow-right:before {
  content: "\f061";
  color: var(--color06);
  padding: 0.5rem;
  border: 0.2rem solid var(--color06);
  border-radius: 10rem;
}
.botao-chamado {
  position: fixed;
  right: 20px;
  bottom: 20px;
  background-color: var(--color03);
  color: #000;
  font-size: 1.5rem;
```

```
        font-weight: bold;
        padding: 0.7rem 1rem; /* Padding inicial */
        border-radius: 2rem;
        box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);
        display: flex;
        align-items: center;
        justify-content: center;
        cursor: pointer;
        text-decoration: none;
        z-index: 1000;
        transition: background-color 0.3s ease, transform 0.3s ease;
/* Transições suaves */
      }
      /* From Uiverse.io by vinodjangid07 */
      .Btn {
        display: flex;
        align-items: center;
        justify-content: flex-start;
        width: 45px;
        height: 45px;
        border: none;
        border-radius: 50%;
        cursor: pointer;
        position: fixed; /* Mudei para fixed */
        bottom: 20px; /* Distância do fundo */
        right: 20px; /* Distância da direita */
        overflow: hidden;
        transition-duration: .3s;
        box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.199);
        background-color:var(--color03);
        color: #000;
        z-index: 1000; /* Garante que o botão fique sobre outros
elementos */
      }


      /* plus sign */
      .sign {
        width: 100%;
        transition-duration: .3s;
        display: flex;
        align-items: center;
        justify-content: center;
      }

      .sign svg {
        width: 17px;
      }
```

```
.sign svg path {
  fill: rgb(0, 0, 0);
}
/* text */
.text {
  position: absolute;
  right: 0%;
  width: 0%;
  opacity: 0;
  color: rgb(0, 0, 0);
  font-size: 1.2em;
  font-weight: 600;
  transition-duration: .3s;
}
/* hover effect on button width */
.Btn:hover {
  width: 125px;
  border-radius: 40px;
  transition-duration: .3s;
}

.Btn:hover .sign {
  width: 30%;
  transition-duration: .3s;
  padding-left: 20px;
}
/* hover effect button's text */
.Btn:hover .text {
  opacity: 1;
  width: 70%;
  transition-duration: .3s;
  padding-right: 10px;
}
/* button click effect*/
.Btn:active {
  transform: translate(2px ,2px);
}

@media (min-width: 700px){
  .container-encaminhamento {
    grid-template-columns: 1fr 1fr;
  }
  .container-header-user_nome{
  font-size: 2rem;
  color: var(--color06);
  }
}

@media (min-width: 1000px){
```

```
        .container-encaminhamento {
          grid-template-columns: 1fr 1fr 1fr;
        }
        #containerBaixo {
          height: 80vh; /* altura ajustada para telas menores */
        }
      }
```

O código JavaScript da tela inicial do Técnico de Manutenção quando carregada inicializa efeitos visuais com o ScrollReveal fazendo surgir os principais elementos da página. Além disso, ele faz chamadas AJAX para buscar e exibir os encaminhamentos da manutenção cujo o técnico está realizando, dividindo-os em "encaminhamentos em andamento" e "pendentes", os mesmos sendo organizados por urgência (alta, média, baixa) e sendo exibidos com cores indicativas.

```
      document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal
        window.sr = ScrollReveal({
            reset: true // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Aplica o efeito de revelação ao header
        sr.reveal('header', {
            duration: 600,
            origin: 'top', // O header surge de cima
            distance: '50px', // Distância que o header vai percorrer
            delay: 200 // Atraso antes de iniciar a animação
        });

        // Aplica o efeito de revelação à parte inferior
        sr.reveal('.container-baixo', {
            duration: 600,
            origin: 'bottom',
            distance: '50px',
            delay: 200
        });

        // Aplica o efeito de revelação ao botão "Solicitar"
        sr.reveal('.Btn', {
            duration: 600,
            origin: 'top', // O botão surge de cima
            distance: '20px', // Distância que o botão vai percorrer
            delay: 400 // Atraso antes de iniciar a animação
```

```
        });
      });

      // Função que retorna os serviços encaminhados para o
funcionário da manutenção que está logado
      function retornaEncaminhamentos() {
        $.ajax({
          url: '/retorna-encaminhamentos',
          type: 'GET',
          success: function(encaminhamentos){
              encaminhamentos_fazendo = encaminhamentos[0]
              encaminhamentos_pendentes = encaminhamentos[1]
              console.log(encaminhamentos[0], encaminhamentos[1])
              document.getElementById('encaminhamentos_fazendo').inne
rHTML = '';
              document.getElementById('encaminhamentos_pendentes').in
nerHTML = '';

              if (encaminhamentos_fazendo.length == 0) {
                document.getElementById('encaminhamentos_fazendo').in
nerHTML = '<h3>Você ainda não aceitou nenhum encaminhamento</h3>';
              }else{
                for (let x = 0; x < encaminhamentos_fazendo.length;
x++) {
                  var    infoEncaminhamentosFazendoYellow    =    `<div
class="ladoEsquerdo-card"
onclick=redirectDetalhes(${encaminhamentos_fazendo[x][3]},${encaminh
amentos_fazendo[x][4]})>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhamentos_fazendo[x][1]}                      |
${encaminhamentos_fazendo[x][0]}</p>
                                    <p                class="urgencia
yellow">${encaminhamentos_fazendo[x][2]} urgência</p>
                                    <a
href="/RF006B/${encaminhamentos_fazendo[x][3]}/${encaminhamentos_faz
endo[x][4]}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                            </div>`;
                  var     infoEncaminhamentosFazendoRed     =     `<div
class="ladoEsquerdo-card">
                            <p class="horario">10:30am</p>
                              <div class="card-conteudo">
                                  <p
class="bloco">${encaminhamentos_fazendo[x][1]}                      |
${encaminhamentos_fazendo[x][0]}</p>
                                  <p                class="urgencia
red">${encaminhamentos_fazendo[x][2]} urgência</p>
```

```
                                    <a
href="/RF006B/${encaminhamentos_fazendo[x][3]}/${encaminhamentos_faz
endo[x][4]}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                              </div>`;
                  var    infoEncaminhamentosFazendoGreen    =    `<div
class="ladoEsquerdo-card">
                              <p class="horario">10:30am</p>
                                <div class="card-conteudo">
                                  <p
class="bloco">${encaminhamentos_fazendo[x][1]}                      |
${encaminhamentos_fazendo[x][0]}</p>
                                  <p                    class="urgencia
green">${encaminhamentos_fazendo[x][2]} urgência</p>
                                  <a
href="/RF006B/${encaminhamentos_fazendo[x][3]}/${encaminhamentos_faz
endo[x][4]}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                              </div>`;

                  //Atualiza o indicador de urgência do serviço que
o Técnico aceitou
                  if(encaminhamentos_fazendo[x][2] === 'media' ){
                    document.getElementById('encaminhamentos_fazend
o').innerHTML += infoEncaminhamentosFazendoYellow;
                  }else if(encaminhamentos_fazendo[x][2] === 'alta'
){
                    document.getElementById('encaminhamentos_fazend
o').innerHTML += infoEncaminhamentosFazendoRed;
                  }else     if(encaminhamentos_fazendo[x][2]     ===
'baixa' ){
                    document.getElementById('encaminhamentos_fazend
o').innerHTML += infoEncaminhamentosFazendoGreen;
                  }
                }
              }

              if (encaminhamentos_pendentes.length == 0) {
                document.getElementById('encaminhamentos_pendentes').
innerHTML = '<h3>Não há encaminhamentos para você</h3>';
              }else{
                for (let x = 0; x < encaminhamentos_pendentes.length;
x++) {
                  var    infoEncaminhamentosPendentesRed    =    `<a
href="/RF006A/${encaminhamentos_pendentes[x][3]}/${encaminhamentos_p
endentes[x][4]}">
                                  <div class="servico-encaminhado">
                                      <div   class="servico-encaminhado-
data">
```

```
                                        <p            class="servico-
encaminhado-dia-semana">${encaminhamentos_pendentes[x][5]}</p>
                                    </div>
                                    <p    class="servico-encaminhado-
local">${encaminhamentos_pendentes[x][1]}                          |
${encaminhamentos_pendentes[x][0]}</p>
                                    <p    class="urgencia-encaminhado
red_pendente">${encaminhamentos_pendentes[x][2]} urgência</p>
                                </div>
                            </a>`;

                var    infoEncaminhamentosPendentesYellow    =    `<a
href="/RF006A/${encaminhamentos_pendentes[x][3]}/${encaminhamentos_p
endentes[x][4]}">
                                <div class="servico-encaminhado">
                                    <div  class="servico-encaminhado-
data">
                                        <p            class="servico-
encaminhado-dia-semana">${encaminhamentos_pendentes[x][5]}</p>
                                    </div>
                                    <p    class="servico-encaminhado-
local">${encaminhamentos_pendentes[x][1]}                          |
${encaminhamentos_pendentes[x][0]}</p>
                                    <p    class="urgencia-encaminhado
yellow_pendente">${encaminhamentos_pendentes[x][2]} urgência</p>
                                </div>
                            </a>`;

                var    infoEncaminhamentosPendentesGreen    =    `<a
href="/RF006A/${encaminhamentos_pendentes[x][3]}/${encaminhamentos_p
endentes[x][4]}">
                                <div class="servico-encaminhado">
                                    <div  class="servico-encaminhado-
data">
                                        <p            class="servico-
encaminhado-dia-semana">${encaminhamentos_pendentes[x][5]}</p>
                                    </div>
                                    <p    class="servico-encaminhado-
local">${encaminhamentos_pendentes[x][1]}                          |
${encaminhamentos_pendentes[x][0]}</p>
                                    <p    class="urgencia-encaminhado
green_pendente">${encaminhamentos_pendentes[x][2]} urgência</p>
                                </div>
                            </a>`;

                    // Atualiza a cor do indicador de urgencia do
ecaminhamento pendente
                    if(encaminhamentos_pendentes[x][2] === 'alta'){
```

```
                    document.getElementById('encaminhamentos_pendentes').innerHTML += infoEncaminhamentosPendentesRed;
                }else    if(encaminhamentos_pendentes[x][2]    === 'media'){
                    document.getElementById('encaminhamentos_pendentes').innerHTML += infoEncaminhamentosPendentesYellow;
                }else    if(encaminhamentos_pendentes[x][2]    === 'baixa'){
                    document.getElementById('encaminhamentos_pendentes').innerHTML += infoEncaminhamentosPendentesGreen;
                }
            }
        }
    },
    error: function(){
        Swal.fire({
            icon: "error",
            title: "Oops...",
            text: "Erro no retorno dos Encaminhamentos!",
            showConfirmButton: false,
            timer: 3500
          });
    }
});
}

// Função para ir para a tela de fazer solicitações
function redirectSolicitacao() {
  window.location.href = '/RF003';
}

// Executa a função retornaEncaminhamentos assim que o documento é carregado
$(document).ready(retornaEncaminhamentos)

// Executa a função retornaEncaminhamentos a cada 5 segundos
setInterval(function(){
  retornaEncaminhamentos()
  console.log('Recarregando...')
}, 5000);

// Função para deslogar da conta
function logout() {
  window.location.href = '/logout';
}

// Função para ir para a tela de detalhes do encaminhamento
function redirectDetalhes(idSolicitacao, idEncaminhamento) {
```

```
        window.location.href                                    =
`/RF006B/${idSolicitacao}/${idEncaminhamento}`;
    }
```

O código Python define duas rotas principais para a tela inicial do tecnico de manutenção. A rota “/RF006” verifica se o usuário está logado, caso autenticado, ela extrai informações como foto e nome completo passando essas informações para a página RF006-TLmanuten.html. Caso o usuário não esteja logado, ele é redirecionado para a página inicial. A função /retorna-encaminhamentos, por sua vez, lida com a consulta dos encaminhamentos de manutenção do técnico dividindo-os em duas categorias: "pendentes" e "em andamento".

```
    # Criando a rota para a tela inicial dos técnicos da manutenção
    @app.route("/RF006")
    def pg_manutencao(): # Função que renderiza a tela inicial dos
técnicos da manutenção
        if "usuario" in session:
            funcao = session["usuario"]["funcao"]
            foto = session["usuario"]["foto"]
            nome_completo = session["usuario"]["nome"]

            # Dividindo o nome em partes
            partes_nome = nome_completo.split()

            # Verificando se o nome tem mais de uma parte
            if len(partes_nome) > 1:
                primeiro_ultimo_nome      =      f"{partes_nome[0]}
{partes_nome[-1]}"
            else:
                # Caso o nome tenha apenas uma parte, retorna
somente essa parte
                primeiro_ultimo_nome = partes_nome[0]

            return          render_template("RF006-TLmanuten.html",
campo_nome = primeiro_ultimo_nome, campo_funcao = funcao, campo_foto
= foto)
        else:
            return redirect("/")

    # Criando a rota que retorna todos os encaminhamentos
    @app.route("/retorna-encaminhamentos")
    def retorna_encaminhamentos(): # Função que retorna todos os
encaminhamentos
```

```
        encaminhamento = Encaminhamento()

        cpf = session["usuario"]["CPF"]

        status = 'à fazer'

        retorna_encaminhamentos_pendentes                                    =
encaminhamento.mostrar_encaminhamentos(status, cpf)

        status = 'fazendo'

        retorna_encaminhamentos_fazendo                                      =
encaminhamento.mostrar_encaminhamentos(status, cpf)

        return            jsonify([retorna_encaminhamentos_fazendo,
retorna_encaminhamentos_pendentes])
```

Apêndice F – Tela Inicial do Solicitante - EASY-REQUEST

Este apêndice contém os códigos responsáveis pela implementação da Tela Inicial do Solicitante. A Tela Inicial do Solicitante inclui cards que contém as informações de todos os serviços encaminhados para os funcionários da manutenção, além de um local que contém as informações básicas do usuário. Os cards são adicionados pelo frontend(JavaScript) utilizando dados extraídos do backend(Python).

Na Tela Inicial do Solicitante, o HTML contém um cabeçalho com as informações básicas do usuário, como nome e função. Em sua parte principal, existe um container vazio que receberá informações do JavaScript e no fim há um container que engloba um botão que redireciona para a página de solicitação.

```
    <!DOCTYPE html>
    <html lang="pt-br">
        <head>
            <meta charset="UTF-8">
            <meta    name="viewport"    content="width=device-width,
initial-scale=1.0">

            <title>Tela Inicial</title>
```

```
            <!-- Link Arquivo CSS Global -->
            <link rel="stylesheet" href="/static/CSS/global.css">
            <!-- Link Arquivo CSS -->
            <link                                    rel="stylesheet"
href="/static/CSS/TLsolicitante.css">
            <!-- Link Arquivo JS -->
            <script                                              defer
src="/static/JS/TLsolicitante.js"></script>
            <!-- Link para os ICONS -->

          <script  src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
          <!-- link Arquivo JS data -->
            <script          defer          src="/static/JS/funcao-
data.js"></script>
            <script     defer     src     =     "/static/JS/funcao-
horas.js"></script>
            <script src="https://unpkg.com/scrollreveal"></script>
            <link                                    rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outl
ined:opsz,wght,FILL,GRAD@24,400,0,0&icon_names=add" />
            <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
          <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
          <!-- link JS biblioteca SweetAlert -->
          <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
          <!-- Link para o usúario receber notificações -->
          <script              defer              type="module"
src="/static/JS/notificacoes.js"></script>

          <!-- Logo do Easy Request na guia-->
          <link     rel="shortcut     icon"     type="imagex/png"
href="/static/IMG/TASK request.svg">

      </head>
      <body>
          <!-- Cabeçalho da página -->
          <header>
              <section class="container-header-user">
                  <div class="info-usuario">
                      <figure><img    src="{{    campo_foto    if
campo_foto                                                  else
'https://st4.depositphotos.com/11574170/25191/v/450/depositphotos_25
1916955-stock-illustration-user-glyph-color-icon.jpg'             }}"
alt="Imagem de Perfil"></figure>
```

```
                        <div      class="container-header-user_info
horas">
                            <h2 class="container-header-user_nome">
<span class="saudacao" id="saudacao"> </span>{{campo_nome}}</h2>
                            <p             class="container-header-
user_cargo">{{campo_funcao}}</p>
                        </div>
                    </div>
                    <button      class="button      button--logout"
onclick="logout()">

                        <div class="button__icon">

                          <svg viewBox="0 0 512 512">
                            <path  d="M377.9  105.9L500.7  228.7c7.2
7.2 11.3 17.1 11.3 27.3s-4.1 20.1-11.3 27.3L377.9 406.1c-6.4 6.4-15
9.9-24 9.9c-18.7 0-33.9-15.2-33.9-33.9l0-62.1-128 0c-17.7 0-32-14.3-
32-32l0-64c0-17.7 14.3-32 32-32l128 0 0-62.1c0-18.7 15.2-33.9 33.9-
33.9c9 0 17.6 3.6 24 9.9zM160 96L96 96c-17.7 0-32 14.3-32 32l0 256c0
17.7 14.3 32 32 32l64 0c17.7 0 32 14.3 32 32s-14.3 32-32 32l-64 0c-
53 0-96-43-96-96L0 128C0 75 43 32 96 32l64 0c17.7 0 32 14.3 32 32s-
14.3 32-32 32z"></path>
                          </svg>

                        </div>


                        <div class="button__text">Sair</div>

                    </button>
                </section>
            </header>

            <!-- Conteúdo principal da página -->
            <main class="container-baixo">
                <section       class="container_baixo-ladoEsquerdo"
id="container_baixo-ladoEsquerdo">

                </section>
            </main>

            <!-- Botão que leva para a tela de solicitação -->
            <section class="container-btn">
                <button                                  class="Btn"
onclick="redirectSolicitacao()">

                  <div class="sign">
                    <span class="material-symbols-outlined">
                        add
```

```
                        </span>

                    <div class="text">Solicitar</div>
                </button>

            </section>
        </body>
    </html>
```

Na Tela Inicial do Solicitante, o CSS tem a função de estilizar os componentes principais da página que são o cabeçalho, o container que abrange os encaminhamentos e o container que abrange o botão de solicitação, tornando assim a página mais atraente e intuitiva. Ele também é responsável por permitir que a página se adapte a diferentes tipos de tela por meio da regra “@media”, que define algumas alterações de layout de acordo com a largura da tela.

```
    body{
        overflow-x: hidden;
    }
    header{
        width: 100vw;
        height: 18vh;
        display: flex;
    }

    /* Estilo do cabeçalho */
    .container-cumprimentacao {
        width: 100vw;
        display: flex;
        justify-content: space-around;
        align-items: center;
    }

    #cumprimentacao{
        align-self: center;
        font-size: 2.5rem;
        font-weight: lighter;
    }
    .info-usuario{
        display: flex;
    }
    /* Estilo da data */
    .container-header-data{
        margin: 1rem;
        display: flex;
```

```
        gap: 0.5rem;
        justify-content: center;
        align-items: center;
        font-size: 1.5rem;
    }
    .data{
        display: flex;
        gap: 0.5rem;
        margin-right: 3rem;
        font-size: 1.8rem;

    }
    .dia-semana{
        color: var(--color05);
        font-weight: 600;
    }

    /* Estilo da Área Usuario */
    .container-header-user{
        width: 100vw;
        display: flex;
        align-items: center;
        padding: 0rem 0rem 0rem 5rem;
        justify-content: space-between;
    }

    .container-header-user img{
        width: 6rem;
        height: 6rem;
        object-fit:cover;
        border-radius: 100%;
    }
    .container-header-user_info {
      margin-left: 1rem;
    }
    .container-header-user_nome{
        font-size: 2rem;
        color: var(--color06);
    }

    .container-header-user_cargo{
        background-color: var(--color05);
        color: var(--color01);
        margin-right: 1rem;
        text-align: center;
        padding: 0.4rem;
        border-radius: 0.5rem;
        font-size: 1.5rem;
    }
```

```css
/* Estilo do css do man do site */
.container-baixo{
  width: 100vw;
  min-height: 72vh;
  max-height: 72vh;
  overflow-y: auto; /* Adiciona rolagem vertical */
  overflow-x: hidden; /* Esconde a rolagem horizontal */
  height: auto;
  background-color: var(--color06);
  margin-bottom: 0;
  position: absolute;
  bottom: 0;
  left: 0;
  border-top-left-radius: 2rem;
  border-top-right-radius: 2rem;
  padding: 1.8rem;
  box-shadow: 0 -4px 10px rgba(20, 23, 53, 0.3);
}

/* barra de rolagem transparente */
::-webkit-scrollbar {
  background-color: transparent;
}

.container_baixo-ladoEsquerdo,.container_baixo-ladoDireito{
    min-height: 50%;
    margin-top: 3rem;
    padding: 1rem;
}

.container_baixo-ladoEsquerdo h3{
  text-align: center;
  color: white;
  grid-column: span 3;
}

h2{
    font-size: 1.8rem;
    font-weight: bold;
    color: var(--color01);
}
hr{
    margin-bottom: 2rem;
}
.destaque-txt{
    color: var(--color03);
}
.ladoEsquerdo-card, .ladoDireito-card{
    background: var(--color04);
```

```
        height: 8rem;
        border-radius: 10px;
        display: flex;
        flex-direction: column;
        justify-content: center;
        font-size: 1.8rem;
        gap: 0.5rem;
        margin-top: 2rem;
        text-transform: capitalize;
        transition-duration: 1s;
    }

    .ladoEsquerdo-conteudo{
      cursor: pointer;
    }

    .card-conteudo{
        display: flex;
        justify-content: space-around;
        align-items: center;
        font-weight: 700;
        transition: transform 0.2s;
    }

    .card-conteudo i:hover{
      transform: scale(1.1);
      color: var(--color04);
      border: var(--color04);
    }

    /* Cores na mudança do card de acordo com a urgencia */
    .red:hover{
      color: var(--color04);
      background-color: var(--colorRed);
    }

    .green:hover{
      background-color: var(--colorGreen);
      color:var(--color04);
    }

    .yellow:hover{
      background-color: var(--color03);
      color:var(--color04);
    }

    .horario{
        padding-left: 1.4rem;
    }
```

```
.urgencia-ind{
  padding: 0.1rem 0.5rem;
  border: 0.2rem solid var(--color02);
  border-radius: 0.5rem;
  color: var(--color02);
  cursor: default;
}

/*Cores de mudança do indicador de urgencia  */
.urgencia-yellow{
  border: 0.2rem solid var(--color04);
  border-radius: 0.5rem;
  color: var(--color01);
  background-color:var(--color03) ;
}

.urgencia-red{
border: 0.2rem solid var(--color04);
border-radius: 0.5rem;
color: var(--color01);
background-color:var(--colorRed) ;
}

.urgencia-green{
border: 0.2rem solid var(--color04);
border-radius: 0.5rem;
color: var(--color01);
background-color:var(--colorGreen) ;
}

.fa-arrow-right:before {
    content: "\f061";
    color: var(--color06);
    padding: 0.5rem;
    border: 0.2rem solid var(--color06);
    border-radius: 10rem;
}
.botao-chamado {
    position: fixed;
    right: 20px;
    bottom: 20px;
    background-color: var(--color03);
    color: #000;
    font-size: 1.5rem;
    font-weight: bold;
    padding: 0.7rem 1rem; /* Padding inicial */
    border-radius: 2rem;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);
    display: flex;
```

```
          align-items: center;
          justify-content: center;
          cursor: pointer;
          text-decoration: none;
          z-index: 1000;
          transition:  background-color  0.3s  ease,  transform  0.3s
ease; /* Transições suaves */
      }
      /* From Uiverse.io by vinodjangid07 */
      .Btn {
          display: flex;
          align-items: center;
          justify-content: flex-start;
          width: 45px;
          height: 45px;
          border: none;
          border-radius: 50%;
          cursor: pointer;
          position: fixed; /* Mudei para fixed */
          bottom: 20px; /* Distância do fundo */
          right: 20px; /* Distância da direita */
          overflow: hidden;
          transition-duration: .3s;
          box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.199);
          background-color:var(--color03);
          color: #000;
          z-index:  1000; /* Garante  que  o  botão  fique  sobre  outros
elementos */
      }


        /* plus sign */
        .sign {
          width: 100%;
          transition-duration: .3s;
          display: flex;
          align-items: center;
          justify-content: center;
        }

        .sign svg {
          width: 17px;
        }

        .sign svg path {
          fill: rgb(0, 0, 0);
        }
        /* text */
        .text {
```

```
      position: absolute;
      right: 0%;
      width: 0%;
      opacity: 0;
      color: rgb(0, 0, 0);
      font-size: 1.2em;
      font-weight: 600;
      transition-duration: .3s;
    }
    /* hover effect on button width */
    .Btn:hover {
      width: 125px;
      border-radius: 40px;
      transition-duration: .3s;
    }

    .Btn:hover .sign {
      width: 30%;
      transition-duration: .3s;
      padding-left: 20px;
    }
    /* hover effect button's text */
    .Btn:hover .text {
      opacity: 1;
      width: 70%;
      transition-duration: .3s;
      padding-right: 10px;
    }
    /* button click effect*/
    .Btn:active {
      transform: translate(2px ,2px);
    }

  .button {
      display: flex;
      align-items: center;
      justify-content: flex-start;
      width: 4rem; /* Largura inicial do botão */
      height: 4rem;
      border: none;
      border-radius: 50%;
      margin: auto;
      cursor: pointer;
      position: relative;
      overflow: hidden;
      transition-duration: .3s;
      box-shadow: 2px 2px 10px rgba(0, 0, 0, 0.199);
      background-color: var(--color05);
      margin-right: 1.5rem;
```

```
}

.button:hover {
  width: 10rem; /* Limita o tamanho no hover para 10rem */
  border-radius: 40px;
  transition-duration: .3s;
}

.button__icon {
  width: 100%;
  transition-duration: .3s;
  display: flex;
  align-items: center;
  justify-content: center;
}

.button__icon svg {
  width: 1.5rem;
}

.button__icon svg path {
  fill: white;
}

.button__text {
  position: absolute;
  right: 0%;
  width: 0%;
  opacity: 0;
  color: white;
  font-size: 1.2rem;
  font-weight: 600;
  transition-duration: .3s;
}

.button:hover .button__icon {
  width: 26%;
  transition-duration: .3s;
  padding-left: 10px;
}

.button:hover .button__text {
  opacity: 1;
  width: 60%; /* Reduz a expansão do texto */
  transition-duration: .3s;
  padding-right: 1rem; /* Ajusta o espaçamento do texto */
}

.button:active {
```

```
        transform: translate(2px, 2px);
      }
      /* -------------MEDIA---------------- */
      @media (max-width: 926px){
        /* Estilo da Área Usuario */
    .container-header-user{
        width: 100vw;
        display: flex;
        align-items: center;
        padding: 0rem 0rem 0rem 1rem;
        justify-content: space-between;
    }

    }
```

O JavaScript da Tela Inicial do Solicitante tem como principal objetivo criar funcionalidades para os botões por meio das funções “redirectSolicitacao()”, “logout()” e “redirectDetalhes()” e adicionar os encaminhamentos na tela a cada 5 segundos utilizando a função “retorna_encaminhados()” e criar uma animação para quando a tela for carregada.

```
    // Função que redireciona para a tela de solicitação
    function redirectSolicitacao(params) {
        window.location.href = '/RF003';
    }
    document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal e define a configuração global
        window.sr = ScrollReveal({
            reset: true  // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Aplica o efeito de revelação ao elemento com o ID
'container-header'
        sr.reveal('.container-header-user', {
            duration: 500,  // Duração da animação em milissegundos
            origin: 'top',   // Elemento surge do topo
            distance: '50px' // Distância percorrida pelo elemento
        });

        // Aplica o efeito de revelação ao elemento com o ID
'conteudo-principal'
        sr.reveal('.container-baixo', {
            duration: 800,  // Duração da animação (mais lenta)
            origin: 'bottom',  // Elemento surge de baixo
```

```
            distance: '100px', // Distância que o elemento vai
percorrer
            delay: 200,  // Atraso antes de iniciar a animação
            easing: 'ease-in-out',   // Tipo de suavização do
movimento
        });
    });

    // Função para deslogar da conta
    function logout() {
        window.location.href = '/logout';
    }

    // Função que mostra todos os encaminhamentos que ainda não
foram finalizados
    function retorna_encaminhados(){
        $.ajax({
            url: '/retorna-encaminhados',
            type: 'GET',
            success: function(encaminhado){
                document.getElementById('container_baixo-
ladoEsquerdo').innerHTML = `<div class="ladoEsquerdo-txt">
                        <h2>Serviços já<span class="destaque-txt">
Encaminhados</span></h2>
                        <hr>
                    </div>`;
                if (encaminhado.length == 0) {
                    document.getElementById('container_baixo-
ladoEsquerdo').innerHTML += '<h3>Não existem encaminhamentos em
aberto</h3>';
                } else {
                    for (let x = 0; x < encaminhado.length; x++) {
                        var infoEncaminhadosYellow =`
                        <div class="ladoEsquerdo-conteudo"
onclick=redirectDetalhes(${encaminhado[x]['id_solicitacao']},${encam
inhado[x]['id_encaminhamento']})>
                            <div class="ladoEsquerdo-card yellow">
                                <p class="horario">|
${encaminhado[x]['status']}</p>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhado[x]['id_sala']}|${encaminhado[x]['bloco']
}</p>
                                    <p class="urgencia-ind
urgencia-yellow" id="urgencia-ind">${encaminhado[x]['urgencia']}</p>
                                    <a href="/detalhes-
encaminhamento/${encaminhado[x]['id_solicitacao']}/${encaminhado[x][
'id_encaminhamento']}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
```

```
                            </div>
                        </div>`;

                        var infoEncaminhadosRed =`
                        <div          class="ladoEsquerdo-conteudo"
onclick=redirectDetalhes(${encaminhado[x]['id_solicitacao']},${encam
inhado[x]['id_encaminhamento']})>
                            <div class="ladoEsquerdo-card red">
                                <p               class="horario">|
${encaminhado[x]['status']}</p>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhado[x]['id_sala']}|${encaminhado[x]['bloco']
}</p>
                                    <p          class="urgencia-ind
urgencia-red" id="urgencia-ind">${encaminhado[x]['urgencia']}</p>
                                    <a             href="/detalhes-
encaminhamento/${encaminhado[x]['id_solicitacao']}/${encaminhado[x][
'id_encaminhamento']}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                            </div>
                        </div>`;

                        var infoEncaminhadosGreen =`
                        <div          class="ladoEsquerdo-conteudo"
onclick=redirectDetalhes(${encaminhado[x]['id_solicitacao']},${encam
inhado[x]['id_encaminhamento']})>
                            <div class="ladoEsquerdo-card green">
                                <p               class="horario">|
${encaminhado[x]['status']}</p>
                                <div class="card-conteudo">
                                    <p
class="bloco">${encaminhado[x]['id_sala']}|${encaminhado[x]['bloco']
}</p>
                                    <p          class="urgencia-ind
urgencia-green" id="urgencia-ind">${encaminhado[x]['urgencia']}</p>
                                    <a             href="/detalhes-
encaminhamento/${encaminhado[x]['id_solicitacao']}/${encaminhado[x][
'id_encaminhamento']}"><i class="fa-solid fa-arrow-right"></i></a>
                                </div>
                            </div>
                        </div>`;

                        // Atualiza o status do card de acordo com
o status do item
                        if (encaminhado[x]['urgencia'] === 'media')
{
                            document.getElementById('container_baix
o-ladoEsquerdo').innerHTML += infoEncaminhadosYellow;
```

```
                        } else if (encaminhado[x]['urgencia'] ===
'alta') {
                            document.getElementById('container_baix
o-ladoEsquerdo').innerHTML += infoEncaminhadosRed;
                        } else if (encaminhado[x]['urgencia'] ===
'baixa') {
                            document.getElementById('container_baix
o-ladoEsquerdo').innerHTML += infoEncaminhadosGreen;
                        };

                    }
                }
            }
        })
    }

    // Executa a função retorna_encaminhamentos assim que o
documento é carregado
    $(document).ready(retorna_encaminhados)

    // Executa a função retorna_encaminhamentos a cada 5 segundos
    setInterval(() => {
        retorna_encaminhados()
        console.log("Recarregado!")
    }, 5000);

    // Função para ir para a tela de detalhes do encaminhamento
    function redirectDetalhes(id_solicitacao, id_encaminhamento) {
        window.location.href = `/detalhes-
encaminhamento/${id_solicitacao}/${id_encaminhamento}`;
    }
```

No BackEnd, o Python(por meio do Flask) renderiza a página e adiciona o nome e função do usuário no HTML utilizando a rota “/tl-solicitante”. Já pela rota chamada “/retorna-encaminhados”, o Python executa a função “mostrar_encaminhados()” com o objetivo de coletar no Banco de Dados as informações relacionadas aos encaminhamentos, para posteriormente essas informações serem adicionadas ao HTML por meio do JavaScript.

```
    # Criando a rota para a tela do administrador
    @app.route("/tl-administrador")
    def tl_administrador(): # Função que renderiza a tela do
administrador
        if "usuario" in session:
            funcao = session["usuario"]["funcao"]
```

```
            nome_completo = session["usuario"]["nome"]

            # Dividindo o nome em partes
            partes_nome = nome_completo.split()

            # Verificando se o nome tem mais de uma parte
            if len(partes_nome) > 1:
                primeiro_ultimo_nome      =      f"{partes_nome[0]}
{partes_nome[-1]}"
            else:
                # Caso o nome tenha apenas uma parte, retorna
somente essa parte
                primeiro_ultimo_nome = partes_nome[0]

            return          render_template("TLadministrador.html",
campo_nome = primeiro_ultimo_nome, campo_funcao = funcao)
        else:
            return redirect("/")
```

```
    # Criando a rota para retornar os serviços que foram
encaminhados para algum funcionário
    @app.route("/retorna-encaminhados")
    def retorna_encaminhados(): # Função que retorna os serviços
que foram encaminhados para algum funcionário
        encaminhamento = Encaminhamento()
        encaminhado = encaminhamento.mostrar_encaminhados()
        return jsonify(encaminhado), 200
```

```
        # Mostra todos os encaminhamentos que ainda não foram
finalizados
        def mostrar_encaminhados(self):
            myBD = Connection.conectar()

            mycursor = myBD.cursor()

            mycursor.execute(f"SELECT      s.id_sala,      s.bloco,
enc.urgencia, enc.status, sol.id_solicitacao, enc.id_encaminhamento
FROM tb_funcionarios func , tb_encaminhamentos enc, tb_solicitacoes
sol, tb_salas s WHERE enc.id_solicitacao = sol.id_solicitacao AND
sol.id_sala     =     s.id_sala     AND     enc.CPF_funcionario     =
func.CPF_funcionario   AND   enc.status_finalizacao   =   FALSE   AND
(enc.status  =  'fazendo'  OR  enc.status  =  'à  fazer')  ORDER  BY
id_encaminhamento DESC;")

            mostrar_encaminhados= mycursor.fetchall()

            lista_encaminhado = []
            for encaminhado in mostrar_encaminhados:
                lista_encaminhado.append({
```

```
                    "id_sala":encaminhado[0],
                    "bloco":encaminhado[1],
                    "urgencia":encaminhado[2],
                    "status":encaminhado[3],
                    "id_solicitacao":encaminhado[4],
                    "id_encaminhamento":encaminhado[5]
                })

            return lista_encaminhado
```

Apêndice G – Tela de Solicitação - EASY-REQUEST

A tela de solicitação é disponibilizada para todos os colaboradores, permitindo o preenchimento de um formulário detalhado, estruturado pelo código HTML, para registrar o tipo de problema identificado, o local de ocorrência e uma descrição da situação.

```
    <!DOCTYPE html>
    <html lang="pt-br">
        <head>
            <meta charset="UTF-8">
            <meta    name="viewport"    content="width=device-width,
initial-scale=1.0">
            <title>Solicitação</title>
            <!-- Estilos CSS: Importação do CSS global e específico
da página -->
            <link rel="stylesheet" href="/static/CSS/global.css">
            <link     rel="stylesheet"     href="/static/CSS/RF003-
solic.css">
            <!-- link JS biblioteca scrollrevealjs -->
            <script src="https://unpkg.com/scrollreveal"></script>
            <!-- Scripts JavaScript: Carregamento dos arquivos de
script da página -->
            <script defer src="/static/JS/RF003-solic.js"></script>
            <script          defer          src="/static/JS/funcao-
voltar.js"></script>
            <!-- Biblioteca de ícones do Font Awesome para uso de
ícones na página -->
            <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
            <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
            <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
```

```
            <!-- link JS biblioteca SweetAlert -->
            <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
            <!-- Link para o usúario receber notificações -->
            <script              defer              type="module"
src="/static/JS/notificacoes.js"></script>

            <!-- Logo do Easy Request na guia-->
            <link     rel="shortcut     icon"     type="imagex/png"
href="/static/IMG/TASK request.svg">
        </head>

        <body>
            <!-- Cabeçalho: Informações do usuário e data -->
            <header>
                <!-- Contêiner do cabeçalho com botão de voltar e
data atual -->
                <section   class="container-header"   id="container-
header">
                    <div class="container-header_btn">
                        <button class="container-header_btn-voltar"
onclick="funcVoltar('solicitacao')"><i    class="fa-solid    fa-arrow-
left"></i>Voltar</button>
                    </div>
                </section>

            </header>

            <!-- Conteúdo principal da página -->
            <main>
                <section    class="container-main"    id="container-
main">
                    <!-- Título e subtítulo da área de solicitação
-->
                    <div class="main-titulo-subtitulo">
                        <h1>Área de <p>Solicitação</p></h1>
                        <p class="subtitulo">Coloque as Informações
sobre o Serviço...</p>
                    </div>
                    <!-- Formulário de solicitação de serviço -->
                    <div class="container-main_form">
                        <!--  Seleção  de  opções  como  tipo  de
serviço, bloco e sala -->
                        <div class="form_select">
                            <select  name="tipo-servico"  id="tipo-
servico">
                                <option      value="">Tipo      de
Serviço</option>
                            </select>
```

```
                            <select name="bloco" id="bloco">
                                <option value="0">Blocos</option>
                            </select>
                            <select name="sala" id="sala">
                                <option value="0">Sala</option>
                            </select>
                             <!-- Upload de arquivo (opcional) -->
                            <div class="form_input-arquivo">
                                <label           for="arquivo">Foto
(Opcional):</label>
                                <input  type="file"  name="arquivo"
id="arquivo">
                            </div>
                        </div>
                        <!--  Campo  de  texto  para  informações
adicionais -->
                        <div class="form_descricao">
                            <label
for="descricao">Descrição:</label>
                            <textarea             name="descricao"
id="descricao"    placeholder="Descreva    o    serviço    a    ser
realizado..."></textarea>
                            <!-- <input type="text" id="descricao"
name="descricao"    placeholder="Descreva    o    serviço    a    ser
realizado..."> -->
                        </div>
                        <!-- Botão de envio da solicitação -->
                        <button    type="submit"    class="form_btn-
enviar" onclick="fazerSolicitacao()">Enviar Solicitação</button>
                    </div>
                </section>
            </main>
        </body>
    </html>
```

O código CSS formata a tela para garantir um formulário intuitivo, responsivo e fácil de preencher. Ele importa a fonte *Montserrat* para a tipografia, utiliza *flexbox* para organizar os elementos e aplicar estilos específicos aos campos do formulário. Além disso, ajusta o layout para diferentes tamanhos de tela.

```
    /* Importa a fonte Montserrat do Google Fonts */
    @import
url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0
,100..900;1,100..900&display=swap');
```

```css
/* Estilo do corpo da página, incluindo a barra de rolagem */
body {
    scrollbar-width: thin;
    scrollbar-color: transparent transparent;
    overflow-x: hidden;
}

/* Estilo da barra de rolagem no WebKit (Chrome, Safari) */
body::-webkit-scrollbar {
    width: 8px;
}

body::-webkit-scrollbar-thumb {
    background-color: rgba(0, 0, 0, 0);
}

body::-webkit-scrollbar-track {
    background-color: rgba(0, 0, 0, 0);
}

/* Define a fonte Montserrat para inputs e selects */
input, select {
    font-family: "Montserrat", sans-serif;
}

/* Estilo do cabeçalho da aplicação */
.container-header {
    width: 100vw;
    height: 20vh;
}

/* Estilo do botão "voltar" no cabeçalho */
.container-header_btn-voltar {
    margin: 1rem;
    padding: 0.5rem;
    width: 10rem;
    border-color: var(--color02);
    border-radius: 5rem;
    background: none;
    font-weight: 700;
    transition-duration: 0.2s;
}

/* Estilo do ícone dentro do botão "voltar" */
.container-header_btn-voltar i {
    margin-right: 0.5rem;
}

/* Estilo do botão "voltar" ao passar o cursor */
```

```
.container-header_btn-voltar:hover {
    border-color: none;
    background-color: var(--color02);
    color: var(--color01);
}

/* Estilo da área principal da solicitação */
.container-main {
    width: 100vw;
    height: 80vh;
    overflow-y: auto;
    background-color: var(--color06);
    color: var(--color01);
    display: flex;
    align-items: center;
    flex-direction: column;
    border-top-left-radius: 2rem;
    border-top-right-radius: 2rem;
    box-shadow: 0 -4px 8px rgba(20, 23, 53, 0.3);
}

/* Estilo do título e subtítulo na área principal */
.main-titulo-subtitulo {
    text-align: center;
    margin-bottom: 1rem;
}

/* Estilo do subtítulo */
.subtitulo {
    font-weight: 100;
    font-size: 1.5rem;
}

/* Estilo do texto dentro do título principal */
.container-main h1 p {
    margin-left: 1rem;
    font-weight: 700;
}

/* Estilo do título principal */
.container-main h1 {
    margin: 1.5rem;
    font-size: 3rem;
    font-weight: 100;
    display: flex;
    justify-content: center;
}

/* Estilo do formulário de envio de solicitação */
```

```css
.container-main_form {
    text-align: center;
}

/* Estilo para o contêiner de seleções (select) */
.form_select {
    display: flex;
    flex-direction: column;
    margin: 0.5rem;
    gap: 2.5rem;
}

/* Estilo específico para os elementos select */
.form_select select {
    background-color: var(--color04);
    color: var(--color06);
    font-weight: 700;
    border: none;
    padding: 0.7rem;
    border-radius: 2rem;
}

/* Estilo para o contêiner de input de arquivo */
.form_input-arquivo {
    display: flex;
    flex-direction: column;
    text-align: left;
    margin-bottom: 2rem;
}

/* Estilo do rótulo do input de arquivo */
.form_input-arquivo label {
    font-size: 1.5rem;
}

/* Estilo do input de arquivo */
.form_input-arquivo input {
    background-color: var(--color04);
    padding: 0.5rem;
    color: var(--color06);
    border-radius: 2rem;
    margin-top: 1rem;
}

/* Estilo específico para o input de arquivo */
#arquivo {
    background-color: var(--color01);
    border: none;
}
```

```
/* Estilo para a descrição do formulário */
.form_descricao {
    display: flex;
    flex-direction: column;
    text-align: left;
    margin: 0.5rem;
    color: var(--color04);
}

/* Estilo do rótulo da descrição */
.form_descricao label {
    font-size: 1.5rem;
}

/* Estilo do campo de descrição */
#descricao {
    width: 100%;
    height: 8rem;
    border-radius: 2rem;
    margin-top: 1rem;
    padding: 1rem;
    resize: none;
}

/* Oculta a barra de rolagem no campo de descrição */
#descricao::-webkit-scrollbar {
    display: none;
}

/* Estilo do botão "Enviar Solicitação" */
.form_btn-enviar {
    display: inline-block;
    width: 20rem;
    padding: 1.2rem;
    text-align: center;
    border-radius: 50px;
    font-size: 1.6rem;
    font-weight: bold;
    text-decoration: none;
    transition: all 0.3s ease-in-out;
    cursor: pointer;
    background-color: var(--color03);
    border: 2px solid var(--color03);
    color: var(--color01);
    margin: 0.5rem 0rem 1rem 0rem;
}

/* Mudança de estilo do botão "Enviar" ao passar o cursor */
```

```
.form_btn-enviar:hover {
    background-color: transparent;
    color: var(--color03);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.02);
    border: 2px solid var(--color03);
}

/* Responsividade para tamanhos de tela maiores */
@media (min-width:376px) {
    .container-main h1 {
        font-size: 2.5rem;
    }
}

@media(min-width:400px) {
    .form_btn-enviar {
        margin: 3rem 0rem 1rem 0rem;
    }
}

@media(min-width:1000px) {
    .container-main h1 {
        font-size: 4.6rem;
    }
    .subtitulo {
        font-size: 2.1rem;
    }
    .container-main_form {
        width: 55%;
    }

    .form_select select {
        font-size: 1.5rem;
    }

    .form_descricao label {
        font-size: 1.6rem;
    }

    #descricao {
        font-size: 1.5rem;
    }

    .form_btn-enviar {
        margin: 2rem 0rem 1rem 0rem;
    }
}
```

O código Javascript da tela de solicitação, utiliza uma animação inicial com a biblioteca ScrollReveal, criando efeitos visuais quando a pagian é iniciada.Além disso, o código coleta informações do formulário, validando os campos obrigatórios para garantir que o usuário preencha todos os dados necessários. Após essa validação, uma solicitação é enviada ao servidor por meio de uma requisição AJAX, incluindo detalhes como tipo de serviço, bloco, sala, descrição e uma imagem. Caso a solicitação seja bem-sucedida, o sistema busca um token de administrador para enviar notificações, garantindo que novas solicitações sejam informadas ao responsável.

O código também utiliza mensagens visuais de feedback para informar o usuário sobre erros ou sucessos durante as interações, com auxílio da biblioteca SweetAlert.

```
     document.addEventListener('DOMContentLoaded', function() {
         // Inicializa o ScrollReveal e define a configuração global
         window.sr = ScrollReveal({
             reset: true  // Efeito se repete sempre que o elemento
reaparece na tela
         });

         // Aplica o efeito de revelação ao elemento com a class
'.container-header'
         sr.reveal('.container-header', {
             duration: 500,
             origin: 'top',
             distance: '50px'
         });
          // Aplica o efeito de revelação ao elemento com o ID
'container-header-user'
          sr.reveal('#container-header-user', {
             duration: 500,
             origin: 'top',
             distance: '50px'
         });

         // Aplica o efeito de revelação ao  elemento com o ID
'container-main'
         sr.reveal('#container-main', {
```

```
            duration: 500,
            origin: 'bottom',
            distance: '100px',
            easing: 'ease-in-out',
            opacity: 0
        });

    });

    // Coleta de elementos do formulário
    inputServico = document.getElementById('tipo-servico');
    inputBloco = document.getElementById('bloco');
    inputSalas = document.getElementById('sala');
    inputDescricao = document.getElementById('descricao');
    inputFoto = document.getElementById('arquivo');

    // Função que usa os dados do formulário para cadastrar uma solicitação
    function fazerSolicitacao() {

        if (inputServico.value == '' || inputSalas == '0' || inputDescricao == '') {
            Swal.fire({
                icon: "error",
                title: "Oops...",
                text: "Você deve preencher todos os campos obrigatórios!",
                showConfirmButton: false,
                timer: 3500
              });
              return;
        }

        var dados = new FormData(); // Cria um novo FormData

        // Adiciona os dados ao FormData
        dados.append('id_servico', inputServico.value);
        dados.append('id_sala', inputSalas.value);
        dados.append('descricao', inputDescricao.value);
        dados.append('foto', inputFoto.files[0]); // Adiciona a foto

        $.ajax({
            url: '/fazer_solicitacao',
            type: 'POST',
            data: dados,
            contentType: false, // Importante para enviar arquivos
            processData: false, // Não processar os dados
            success: function(){
```

```
                // Obter o token do administrador
                $.ajax({
                    url: '/token-administrador',
                    type: 'GET',
                    success: function(token){
                        if(token['permissao'] == 'administrador'){
                            return;
                        }else{
                            var tokenAdministrador = token['token_administrador'][0];  // Aqui você obtém o token do administrador
                        }

                            // Agora que temos o token, podemos chamar a função para enviar a notificação
                            enviarNotificacaoParaAdministrador(tokenAdministrador);
                    },
                    error: function(){
                        swal("Oops!", "Erro no retorno do Token!", "error");
                    }
                });

                // Função para enviar a notificação para o administrador
                function enviarNotificacaoParaAdministrador(tokenAdministrador) {
                    fetch('/enviar-notificacao/solicitacao', {
                        method: 'POST',
                        headers: {
                            'Content-Type': 'application/json'
                        },
                        body: JSON.stringify({
                            token: tokenAdministrador,
                            mensagem: "Uma nova solicitação de serviço foi realizada!"  // Mensagem que você deseja enviar ao administrador
                        })
                    })
                    .then(response => response.json())
                    .then(data => {
                        console.log("Notificação enviada com sucesso:", data);
                    })
                    .catch((error) => {
                        console.error("Erro ao enviar notificação:", error);
                    });
```

```
                }

                // Exibir a mensagem de sucesso para o usuário
                Swal.fire({
                    position: "center",
                    icon: "success",
                    title: "Solicitação Enviada!",
                    showConfirmButton: false,
                    timer: 1200
                });

                setTimeout(() => {
                    window.location.href = '/RF003';
                }, 1500);
            },
            error: function(){
                Swal.fire({
                    icon: "error",
                    title: "Oops...",
                    text: "Erro ao fazer solicitação!",
                    showConfirmButton: false,
                    timer: 3500
                });
            }
        });

    }

    // Função que coloca os tipos de serviço no campo do formulário
    function retornaServicos() {
        $.ajax({
            url: '/retorna_servicos',
            type: 'GET',
            success: function(servicos){
                for (let x = 0; x < servicos.length; x++) {
                    var option = document.createElement('option');
                    option.value = servicos[x][0];
                    option.textContent = servicos[x][1];
                    inputServico.append(option);
                }
            },
            error: function(){
                swal ( "Oops!" ,  "Erro no retorno do Serviço!" ,
"error" );
            }
        });
    }
```

```
    // Função que retorna todos os blocos do SENAI no campo do formulário
    function retornaBlocos() {
        $.ajax({
            url: '/retorna_blocos',
            type: 'GET',
            success: function(blocos){
                for (let x = 0; x < blocos.length; x++) {
                    var option = document.createElement('option');
                    option.value = blocos[x][0];
                    option.textContent = blocos[x][0];
                    inputBloco.append(option);
                }
            },
            error: function(){
                swal ( "Oops!" ,  "Erro no retorno dos blocos!" , "error" );
            }
        });
    }

    // Função que retorna todas as salas do SENAI no campo do formulário
    function retornaSalas(bloco) {
        inputSalas.innerHTML = '<option value="0">Sala</option>';
        $.ajax({
            url: `/retorna_salas/${bloco}`,
            type: 'GET',
            success: function(salas){
                for (let x = 0; x < salas.length; x++) {
                    var option = document.createElement('option');
                    option.value = salas[x][0];
                    option.textContent = salas[x][1] + ' - ' + salas[x][0];
                    inputSalas.append(option);
                }
            },
            error: function(){
                swal ( "Oops!" ,  "Erro no retorno das Salas!" , "error" );
            }
        });
    }

    // Executando todas as funções
    retornaServicos();
    retornaBlocos();
    retornaSalas();
```

```
    // Executando a função retornaSalas somente quando o campo bloco for alterado
    inputBloco.addEventListener('change', function() {
        retornaSalas(inputBloco.value)
    });
```

O código Python implementa uma Rota para a tela de solicitação que verifica se o usuário está logado pela sessão. Caso o usuário tenha uma sessão ativa, ela coleta informações e manda para a página RF003-solic.html. Assim tambem, a função, fazer_solicitacao(), é responsável por receber os dados do formulário de solicitação, como o tipo de serviço, a sala, a descrição e a foto, e a função então chama o método solicitar_servico da classe Solicitacao para registrar a solicitação no banco de dados. Dependendo do sucesso ou falha da operação, uma resposta JSON é retornada com o status da solicitação. Uma Rota para retornar serviços disponíveis (/retorna_servicos) que consulta o banco de dados para obter todos os serviços disponíveis, retornando os resultados em formato JSON. A função retorna_blocos() faz uma consulta SQL para obter todos os blocos disponíveis, retornando-os também em formato JSON. A função retorna_salas(bloco) retorna todas as salas de um determinado bloco ou todas as salas.

```
    # Criando rota para a tela de solicitacao
    @app.route("/RF003")
    def pg_solicitacao(): # Função que abre a tela de fazer solicitação
        if "usuario" in session:
            funcao = session["usuario"]["funcao"]
            foto = session["usuario"]["foto"]
            nome_completo = session["usuario"]["nome"]

            # Dividindo o nome em partes
            partes_nome = nome_completo.split()

            # Verificando se o nome tem mais de uma parte
            if len(partes_nome) > 1:
                primeiro_ultimo_nome = f"{partes_nome[0]} {partes_nome[-1]}"
```

```
            else:
                # Caso o nome tenha apenas uma parte, retorna
somente essa parte
                primeiro_ultimo_nome = partes_nome[0]

            return render_template("RF003-solic.html", campo_nome =
primeiro_ultimo_nome, campo_funcao = funcao, campo_foto = foto)
        else:
            return redirect("/")

    # Criando rota para a função que executa a solicitação
    @app.route("/fazer_solicitacao", methods=["POST"])
    def fazer_solicitacao(): # Função que executa a solicitação
        id_servico = request.form.get('id_servico')
        id_sala = request.form.get('id_sala')
        descricao = request.form.get('descricao')
        foto = request.files.get('foto')
        cpf = session["usuario"]["CPF"]

        if foto:
            link_arquivo_foto = upload_file(foto)
        else:
            link_arquivo_foto = None

        solicitacao = Solicitacao()

        if    solicitacao.solicitar_servico(id_servico,    id_sala,
descricao, cpf, link_arquivo_foto):
            return jsonify({'mensagem':'Cadastro OK'}), 200
        else:
            return {'mensagem':'ERRO'}, 500

    # Criando rota para a função que retorna os serviços possíveis
    @app.route("/retorna_servicos")
    def  retorna_servicos():  #  Função  que  retorna  os  serviços
possíveis
        myBD = Connection.conectar()

        mycursor = myBD.cursor()

        mycursor.execute(f"SELECT * FROM tb_servicos")

        servicos = mycursor.fetchall()

        return jsonify(servicos), 200

    # Criando rota para a função que retorna todos os blocos do
SENAI
    @app.route("/retorna_blocos")
```

```
    def retorna_blocos(): # Função que retorna todos os blocos do
SENAI
        myBD = Connection.conectar()

        mycursor = myBD.cursor()

        mycursor.execute(f"SELECT DISTINCT bloco FROM tb_salas")

        blocos = mycursor.fetchall()

        return jsonify(blocos), 200

    # Criando rota para a função que retorna todas as salas do
SENAI
    @app.route("/retorna_salas/<bloco>")
    def retorna_salas(bloco): # Função que retorna todas as salas
do SENAI
        myBD = Connection.conectar()

        mycursor = myBD.cursor()

        print(f"Bloco: {bloco}")

        if bloco == "0":
            mycursor.execute(f"SELECT   id_sala,   nome_sala   FROM
tb_salas")
        else:
            mycursor.execute(f"SELECT   id_sala,   nome_sala   FROM
tb_salas WHERE bloco = '{bloco}'")

        sala = mycursor.fetchall()

        return jsonify(sala), 200
```

Apêndice H – Administrador Recebe Solicitação

Esse apêndice é responsável pela tela em que o administrador recebe a solicitação de serviço do Easy Request. Essa tela inclui vários cards que lá estão os serviços solicitados, essa tela foi feita em Html como marcação dos códigos, o CSS para estilização da tela, o Java Script que deixa o sistema inteligente e o Python que faz a ligação entre o sistema e o banco de dados

O código HTML nesta página em um container é onde os cards das solicitações de serviços que são agrupados mostrando todas as solicitações que não foram encaminhadas ainda, e se caso o supervisor quiser pesquisar uma solicitação em específico existe um campo input de pesquisa que assim que o supervisor pesquisar vai aparecer a solicitação desejada

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <!-- Link do css do arquivo -->
    <link rel="stylesheet" href="/static/CSS/global.css">
    <link rel="stylesheet" href="/static/CSS/RF004-
ADMrecbSolic.css">
    <!-- Scripts JavaScript: Carregamento dos arquivos de
script da página -->
    <script defer src="/static/JS/funcao-data.js"></script>
    <script defer src = "/static/JS/funcao-horas.js"></script>
    <script defer src="/static/JS/RF004-
ADMrecbSolic.js"></script>
    <!-- Biblioteca de ícones do Font Awesome para uso de
ícones na página -->
    <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
    <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
    <!-- link JS biblioteca SweetAlert -->
    <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <script src="https://unpkg.com/scrollreveal"></script>
    <!-- Link para o usúario receber notificações -->
    <script defer type="module"
src="/static/JS/notificacoes.js"></script>

    <!-- Logo do Easy Request na guia-->
    <link rel="shortcut icon" type="imagex/png"
href="/static/IMG/TASK request.svg">
    <title>Solicitações</title>
</head>
<body>
    <!-- Área de Informação do Usuário -->
```

```
        <header class="header">
            <!-- Container para a data e botão de menu -->
            <div  class="header__date-container"  id="header__date-
container">
                <div class="header__button">
                    <button              class="header_btn-voltar"
onclick="funcVoltar()"><i          class="fa-solid          fa-arrow-
left"></i>Voltar</button>
                </div>
            </div>
        </header>

        <main>
            <!-- Área Inferior do site -->
            <section class="sectionBaixo">
                <p  class="espacoP">Pesquise por:</p>
                <!-- Barra de Pesquisa -->
                <div class="sectionBaixo__search-container">
                    <input  type="search"  id="search"  name="search"
placeholder="Ex:  solicitante,  descrição,  bloco,  sala,  data,  tipo  de
serviço   e   cargo   "   class="sectionBaixo__search-input"   aria-
label="Campo de pesquisa">
                    <button                          type="submit"
class="sectionBaixo__search-button"          aria-label="Pesquisar"
id="btnSearch">Pesquisar</button>
                </div>

                <!-- Seção de Cards -->
                <section         class="sectionBaixo__grid-section"
id="solicitacoes">

                </section>
            </section>
            <script                          src="/static/JS/RF004-
ADMrecbSolic.js"></script>
        </main>
    </body>

    </html>
```

O código CSS desta página está adaptando o contêiner “sectionBaixo__grid-section”

para deixar o local dos cards bem dinâmico e interativo e no input está modificado para o supervisor da manutenção e o “@media” está com a funcionalidade de ajustar o sistema para qualquer tela.

```
            max-width: 1000rem;
            padding: 1rem;
            font-size: 1.2rem;
            border: 1px solid var(--color05);
            border-radius: 5px;
            outline: none;
        /* Importação de fonte */
        @import
url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0
,100..900;1,100..900&display=swap');
        /* Para remover a barra de rolagem horizontal */
        html, body {
            overflow-x: hidden;
        }
        /* Tornar a barra de rolagem vertical transparente */
        body {
            scrollbar-width: thin; /* Para navegadores como Firefox */
            scrollbar-color: transparent transparent; /* Cor da barra e
do trilho */

            display: flex;
            flex-direction: column;
        }

        body::-webkit-scrollbar {
            width: 8px; /* Defina a largura da barra de rolagem */
        }

        body::-webkit-scrollbar-thumb {
            background-color: rgba(0, 0, 0, 0); /* Deixar transparente
*/
        }

        body::-webkit-scrollbar-track {
            background-color: rgba(0, 0, 0, 0); /* Deixar o trilho
transparente */
        }

        /* Estilo do Header */
        .header {
            width: 100vw;
        }

        /* Container da Data e Botão */
        .header__date-container {
            height: 10vh;
            display: flex;
            justify-content: space-between;
            align-items: center;
```

```
    }
    .container-header-user_info {
        margin-left: 1rem;
    }
    .header_btn-voltar{
        margin: 1rem;
        padding: 0.5rem;
        width: 10rem;
        border-color: var(--color02);
        border-radius: 5rem;
        background: none;
        font-weight: 700;
        transition-duration: 0.2s;
        color: var(--color02);
      }

      .header_btn-voltar i{
        margin-right: 0.5rem;
      }

      .header_btn-voltar:hover{
        border-color: none;
        background-color: var(--color02);
        color: var(--color01);
      }

    /* Estilo da SectionBaixo */
    /* ----------------------------------------------------------------
---------------------------------- */
    .sectionBaixo {
        width: 100vw;
        min-height: 75vh;
        max-height: 75vh;
        overflow-y: auto; /* Adiciona rolagem vertical */
        overflow-x: hidden; /* Esconde a rolagem horizontal */
        height: auto;
        background-color: var(--color06);
        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        padding: 1.5rem;
        box-shadow: 1px -3px 10px rgba(0, 0, 0, 0.5);

        /* Posicionamento */
        position: fixed;
        bottom: 0;
        left: 0;

        /* Altura dinâmica: calculada com base no espaço disponível
*/
```

```
        height: 40%; /* Você pode ajustar este valor ou usar % */
        max-height: 40rem; /* Limita a altura máxima */
        overflow-y: auto; /* Habilita a rolagem vertical */
        overflow-x: hidden;
        z-index: 1000;
    }
    .espacoP{
        margin-top: 0.5rem;
        font-size: 1.5rem;
        color: var(--color01);
        margin-left: 1rem;
    }
    /* Estilo da Barra de Pesquisa */
    .sectionBaixo__search-container {
        display: flex;
        align-items: center;
        justify-content: center;
        margin: 1rem;
    }

    .sectionBaixo__search-input {
        width: 100%;
        max-width: 1000rem;
        padding: 1rem;
        font-size: 1.2rem;
        border: 1px solid var(--color05);
        border-radius: 5px;
        outline: none;
    }

    .sectionBaixo__search-button {
        background-color: var(--color05);
        color: var(--color01);
        border: 1px solid var(--color05);
        border-radius: 5px;
        padding: 1rem 1.5rem;
        font-size: 1.3rem;
        cursor: pointer;
        margin-left: 0.5rem;
    }

    .sectionBaixo__search-button:hover {
        background-color: var(--color07);
        color: var(--color05);
    }

    /* CSS Filtro */
    .sectionBaixo__filter-container {
        display: flex;
```

```
        align-items: center;
        justify-content: center;
        margin: 1rem;
    }

    .sectionBaixo__filter-select {
        width: 100%;
        max-width: 1000rem;
        padding: 1rem;
        font-size: 1.2rem;
        border: 1px solid var(--color05);
        border-radius: 5px;
        outline: none;
    }

    /* Grid padrão - 1 coluna para telas pequenas */
    .sectionBaixo__grid-section {
        display: grid;
        grid-template-columns: 1fr;
        gap: 1rem;
        padding: 1rem;
    }

    .sectionBaixo__grid-section h3{
        text-align: center;
        color: white;
        grid-column: span 3;
    }

    /* Estilo para os itens do grid */
    .sectionBaixo__grid-item {
        padding: 1rem;
        text-align: center;
        border-radius: 5px;
    }

    /* Estilo do Card */
    .card {
        background-color: var(--color04);
        border-radius: 1rem;
        padding: 1.5rem;
        max-width: 30rem;
        margin: 1rem auto;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
        position: relative;
    }

    /* Estilo do Header do Card */
    .card__header {
```

```
        background-color: var(--color05);
        color: var(--color01);
        padding: 0.7rem 1.2rem;
        border-top-left-radius: 0.8rem;
        border-bottom-right-radius: 0.5rem;
        font-weight: bold;
        font-size: 1.5rem;
        position: absolute;
        top: -1rem;
        left: 0;
    }

    /* Corpo do Card */
    .card__body {
        margin-top: 2rem;
        text-align: left;
        font-size: 1.6rem;
        color: black;
    }

    .container-card{
        display: flex;
        justify-content:space-between;

    }

    .card__service, .card__responsavel {
        font-weight: bold;
        margin-bottom: 0.5rem;
        color: black;
    }

    .card__descricao {
        margin-top: 1rem;
        line-height: 1.5;
        overflow-wrap: break-word;

        overflow: hidden;
        text-overflow: ellipsis;
        display: -webkit-box;
        -webkit-line-clamp: 2;
        -webkit-box-orient: vertical;
    }

    /* Linha Divisória */
    .card__linha {
        border: none;
        border-top: 0.3rem solid var(--color06);
        margin: 1rem 0;
```

```
}

/* Rodapé do Card */
.card__footer {
    display: flex;
    justify-content: center;
    margin-top: 1.5rem;
}

.card__button {
    background-color: var(--color03);
    color: var(--color02);
    padding: 0.8rem 2rem;
    border: none;
    border-radius: 0.5rem;
    cursor: pointer;
}

.card__button:hover {
    background-color: var(--color05);
    color: var(--color02);
}

.card__button a{
    color: var(--color02);
    text-decoration: none;
}


/* 2 colunas para telas médias */
@media (min-width: 740px) {
    .sectionBaixo__grid-section {
        grid-template-columns: 1fr 1fr;
    }
    .card {
        max-width: 33rem;
    }
}
@media (min-width: 440px) {
    .sectionBaixo {
        width: 100vw;
        min-height: 80vh;
        max-height: 80vh;
    }
}

/* 3 colunas para telas grandes */
@media (min-width: 1030px) {
    .horas h2{
```

```
            font-size: 3.5rem;
        }
        .sectionBaixo__grid-section {
            grid-template-columns: 1fr 1fr 1fr;
        }
        .card {
            max-width: 31.5rem;
        }

        .header__date{
            margin-top: 3rem;
            gap: 0.5rem;
            border-radius: 5rem;
            box-shadow: 4px 3px 10px rgb(135, 135, 135);
            padding: 1rem;
            font-size: 2.5rem;
        }
    }
```

O JavaScript que é responsável para deixar a tela inteligente está com a função do carregamento automático da página e com o campo de pesquisa, e a função em AJAX que faz as solicitações de serviços serem mostradas, e quando dá algum erro ele mostra uma mensagem de erro

```
    document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal e define a configuração global
        window.sr = ScrollReveal({
            reset: true // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Revelar a seção principal
        sr.reveal('.sectionBaixo', { // Mude para a classe
            duration: 500,
            origin: 'bottom',
            distance: '50px',
            opacity: 0
        });

        // Após as solicitações serem carregadas
        mostrarSolicitacoes();
    });

    // Função que mostra as solicitações
    function mostrarSolicitacoes() {
        if (document.querySelector('#search').value == ''){
```

```
            var pesquisa = null;
        }else{
            var pesquisa = document.querySelector('#search').value;
        }
        $.ajax({
            url: `/retorna-solicitacoes/${pesquisa}`,
            type: 'GET',
            success: function(solicitacoes){
                console.log(solicitacoes)
                document.getElementById('solicitacoes').innerHTML =
'';

                if (solicitacoes.length == 0) {
                    document.getElementById('solicitacoes').innerHT
ML = '<h3>Não há solicitações</h3>';
                }else{
                    for (let x = 0; x < solicitacoes.length; x++) {
                        //          var          option          =
document.createElement('option');
                        // option.value = blocos[x][0];
                        // option.textContent = blocos[x][0];
                        // inputBloco.append(option);
                        var     infoSolicitacoes     =     `<article
class="sectionBaixo__grid-item">
                    <div class="card">
                        <!-- Cabeçalho do card com localização do
prédio -->
                        <header class="card__header">
                            <span    class="card__building"    aria-
label="Localização      do      prédio">${solicitacoes[x][2]}      -
${solicitacoes[x][1]}</span>
                        </header>
                        <div class="card__body">
                            <div class="container-card">
                            <!-- Serviço descrito no card -->
                            <p      class="card__service"      aria-
label="Serviço">${solicitacoes[x][5]}</p>
                            <p       class="card_data"       aria-
label="Horário do Serviço">${solicitacoes[x][6]} |</p>
                            </div>
                            <!-- Pessoa que fez a solicitação -->
                            <p    class="card__responsavel"    aria-
label="Responsável pela solicitação">${solicitacoes[x][4]}</p>
                            <!--  Linha  horizontal  para  separação
visual -->
                            <hr      class="card__linha"      aria-
hidden="true">
                            <!-- Descrição do serviço -->
```

```
                            <p     class="card__descricao"     aria-
label="Descrição do serviço">
                                ${solicitacoes[x][3]}
                            </p>
                        </div>
                        <!-- Rodapé do card com botão de ação -->

                        <footer class="card__footer">
                            <a
href="/RF004A/${solicitacoes[x][0]}"><button     class="card__button"
aria-label="Encaminhar solicitação">Detalhes</button></a>
                        </footer>
                    </div>

                    </article>`
                        document.getElementById('solicitacoes').inn
erHTML += infoSolicitacoes;
                    }
                }
            },
            error: function(){
                Swal.fire({
                    icon: "error",
                    title: "Oops...",
                    text: "Erro no retorno das solicitações!",
                    showConfirmButton: false,
                    timer: 3500
                  });
            }
        });
    }

    // Função para voltar para a tela do administrador
    function funcVoltar() {
        window.location.href = '/tl-administrador'
    }

    // Executa a função mostrarSolicitacoes assim que o documento
carregar
    $(document).ready(mostrarSolicitacoes)

    setInterval(function(){
        mostrarSolicitacoes()
        console.log('Recarregando...')
    }, 5000);

    // Funções do botão de pesquisa
    document.addEventListener('DOMContentLoaded', function () {
        var btnPesquisa = document.querySelector('#btnSearch');
```

```
        // Verifica se o elemento #botaoPesquisa existe antes de
adicionar o event listener
        if (btnPesquisa) {
            btnPesquisa.addEventListener('click', function() {
                mostrarSolicitacoes();
            });
        }

        // Adiciona o evento de "keypress" ao campo de entrada, se
necessário
        var inputPesquisa = document.querySelector('#search');
        if (inputPesquisa) {
            inputPesquisa.addEventListener("keypress",     function
(event) {
                if (event.key === "Enter") {
                    mostrarSolicitacoes();
                }
            });
        }

        // Chama a função assim que a página carrega
        mostrarSolicitacoes();
    });
```

O Python tem as funções “mostrar_solicitacoes” e “recebimento_solicitacoes” que faz a ligação entre o banco para o sistema que faz puxar todos os dados das solicitações de serviço para o JavaScript mostrar as solicitações

```
    # Criando rota que retorna as solicitações
    @app.route("/retorna-solicitacoes/<pesquisa>")
    def  mostrar_solicitacoes(pesquisa):  # Função  que  retorna  as
solicitações
        servico = Solicitacao()
        solicitacoes = servico.recebimento_solicitacoes(pesquisa)
        return jsonify(solicitacoes), 200
```

```
     def recebimento_solicitacoes(self, pesquisa):
                self.pesquisa = pesquisa
                myBD = Connection.conectar()
                mycursor = myBD.cursor()

                if pesquisa == "null":
                    pesquisa = ""
```

```
                sql =(f"SELECT sol.id_solicitacao, sv.nome,
id_sala, descricao, f.nome, IF(fn.id_funcao IS NULL,
'Administrador', fn.nome) AS nome_funcao ,
DATE_FORMAT(sol.data_inicio, '%d/%m/%Y') AS data_inicio_brasileira
FROM tb_solicitacoes sol JOIN tb_funcionarios f ON f.CPF_funcionario
= sol.CPF_funcionario LEFT JOIN tb_funcoes fn ON fn.id_funcao =
f.id_funcao JOIN tb_servicos sv ON sv.id_servico = sol.id_servico
LEFT JOIN tb_encaminhamentos enc ON enc.id_solicitacao =
sol.id_solicitacao WHERE (sv.nome LIKE '%{pesquisa}%' OR sol.id_sala
LIKE '%{pesquisa}%' OR descricao LIKE '%{pesquisa}%' OR f.nome LIKE
'%{pesquisa}%' OR fn.nome LIKE '%{pesquisa}%') AND
enc.id_solicitacao IS NULL ORDER BY sol.data_inicio DESC;")
                mycursor.execute(sql)
                recebimento = mycursor.fetchall()

                return recebimento
```

Apêndice I – Tela de Detalhes da Solicitação

Este apêndice abrange a tela de detalhes de solicitação, onde o administrador ao clicar no serviço que foi encaminhado para o técnico de manutenção será possível a visualização dos detalhes que o serviço possui.

O código a seguir determina a estrutura na tela de detalhes da solicitação, o código puxa as informações de outros arquivos - como CSS, Javascript, links de fonte e de bibliotecas virtuais, determina o local do logo, do botão de redirecionamento para a página anterior, título, texto do detalhamento do serviço, imagem, botão de encaminhamento e excluir.

```
    <!DOCTYPE html>
    <html lang="pt-br">
    <head>
        <!-- Definições básicas de meta tags: caracteres e
responsividade -->
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <!-- Importação de arquivos CSS: global e específico desta
página -->
        <link rel="stylesheet" href="/static/CSS/global.css">
        <link rel="stylesheet" href="/static/CSS/RF004A-
detlSolic.css">
```

```
        <!-- Scripts JavaScript: Carregamento dos arquivos de
script da página -->
        <script defer src="/static/JS/RF004A-
detlSolic.js"></script>
        <!-- Biblioteca jQuery para manipulação de DOM e
requisições Ajax -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <!-- Biblioteca Font Awesome para uso de ícones na página -
->
        <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
        <script src="https://unpkg.com/scrollreveal"></script>
        <!-- link JS biblioteca SweetAlert -->
         <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
         <!-- Link para o usúario receber notificações -->
         <script defer type="module"
src="/static/JS/notificacoes.js"></script>

        <!-- Logo do Easy Request na guia-->
        <link rel="shortcut icon" type="imagex/png"
href="/static/IMG/TASK request.svg">
        <title>Detalhes da Solicitação</title>
    </head>
    <body>
        <!-- Navegação principal com botão de voltar -->
        <nav>
            <section class="container-header" id="container-
header">
                <div class="container-header_btn">
                    <button class="container-header_btn-voltar"
onclick="funcVoltar()"><i class="fa-solid fa-arrow-
left"></i>Voltar</button>
                </div>
            </section>
        </nav>
        <!-- Conteúdo principal da página -->
        <main class="conteudo-principal" id="conteudo-principal">
            <section class="detalhes-solicitacao" id="detalhes-
solicitacao">
                <!-- Cabeçalho da solicitação com título e autor --
>
                <div class="detalhes-solicitacao__cabecalho">
                    <h1 class="detalhes-
solicitacao__titulo">{{campo_sala}} - {{campo_servico}}</h1>
```

```
                    <p                                class="detalhes-
solicitacao__autor">{{campo_nome_solicitante}}                        -
{{campo_funcao_solicitante}}</p>
                </div>
                <!-- Linha divisora entre o cabeçalho e o corpo da
solicitação -->
                <hr class="detalhes-solicitacao__divisor">

                <!--  Corpo  da  solicitação  com  a  descrição  do
serviço e botão de encaminhamento -->
                <div class="detalhes-solicitacao__corpo">
                    <p          class="detalhes-solicitacao__rotulo-
descricao">Descrição</p>
                    <p           class="detalhes-solicitacao__texto-
descricao">
                        {{campo_descricao}}
                    </p>
                    {% if campo_foto == None %}
                    {%else%}
                    <img   src="{{campo_foto}}"   alt="Imagem   que
ilustra a solicitação">
                    {% endif %}
                    <div class="container-botao">
                        <!--  Botão  para  encaminhar  a  solicitação,
redirecionando para outra página -->
                    <a
href="/RF005/{{campo_id_solicitacao}}"><button  class="botao  botao-
cadastro">Encaminhar</button></a>
                    <!-- Botão para excluir esta solicitação-->
                    <button        class="botao       botao-excluir"
onclick="deletarSolicitacao({{campo_id_solicitacao}})">Excluir</butt
on>
                    </div>
                </div>
            </section>
        </main>
    </body>
    </html>
```

O CSS estiliza a tela de detalhamento do serviço, adicionando cores, tamanho de divisões da tela e fontes. Além de determinar a mídia, mudando o tamanho dos elementos para serem responsivos a diversos dispositivos.

```
    /* Estilo para o elemento body */
    body {
        overflow-x: hidden;
        height: auto;
```

```
}

/* Estilo para links */
a {
    text-decoration: none;
}

/* Estilo para o elemento nav */
nav {
    width: 100vw;
    height: 9vh;
    display: flex;
    justify-content: space-between;
    align-items: flex-end;
    margin-top: 1.5rem;
}



/* Estilo para botões no cabeçalho */
.container-header_btn {
    display: flex;
    justify-content: center;
    align-items: center;
}

/* Estilo específico para o botão de voltar */
.container-header_btn-voltar {
    margin: 1rem;
    padding: 0.5rem;
    width: 10rem;
    border-color: var(--color02);
    border-radius: 5rem;
    background: none;
    font-weight: 700;
    transition-duration: 0.2s;
}

/* Estilo para ícone dentro do botão de voltar */
.container-header_btn-voltar i {
    margin-right: 0.5rem;
}

/* Mudança de cor do botão ao passar o mouse */
.container-header_btn-voltar:hover {
    border-color: none;
    background-color: var(--color02);
    color: var(--color01);
}
```

```css
/* Estilo para a seção de detalhes da solicitação */
.detalhes-solicitacao {
    width: 100vw;
    min-height: 75vh;
    max-height: 75vh;
    overflow-y: auto; /* Adiciona rolagem vertical */
    overflow-x: hidden; /* Esconde a rolagem horizontal */
    height: auto;
    background-color: var(--color06);
    margin-bottom: 0;
    position: absolute;
    bottom: 0;
    left: 0;
    border-top-left-radius: 2rem;
    border-top-right-radius: 2rem;
    padding: 1.5rem;
    display: flex;
    flex-direction: column;
    align-content: center;
    justify-content: space-evenly;
    box-shadow: 1px -3px 10px rgba(0, 0, 0, 0.5);
    overflow-wrap: break-word;
    overflow-y: auto; /* Habilita a rolagem vertical */
}
/* estilo da barra de rolagem */
::-webkit-scrollbar {
    background-color: transparent;
}

/* Estilo para o cabeçalho da seção de detalhes */
.detalhes-solicitacao__cabecalho {
    text-align: center;
}

/* Estilo para o título na seção de detalhes */
.detalhes-solicitacao__titulo {
    font-size: 3rem;
    color: var(--color05);
    margin-bottom: 0.5rem;
    overflow-wrap: break-word;
}

/* Estilo para o autor da solicitação */
.detalhes-solicitacao__autor {
    font-size: 3.5rem;
    color: var(--color01);
    overflow-wrap: break-word;
}
```

```
    /* Estilo para o divisor na seção de detalhes */
    .detalhes-solicitacao__divisor {
        border: none;
        border-top: 2px solid var(--color05);
        margin: 1.8rem 0;
    }

    /* Estilo para o corpo da seção de detalhes */
    .detalhes-solicitacao__corpo {
        padding: 0 3rem;
        font-size: 2.3rem;
        display: flex;
        flex-direction: column;
        align-items: center;
        text-align: justify;
        overflow-wrap: break-word;
    }

    /* Estilo para o rótulo da descrição */
    .detalhes-solicitacao__rotulo-descricao {
        font-weight: bold;
        color: var(--color01);
        font-size: 2rem;
        overflow-wrap: break-word;
    }

    /* Estilo para o texto da descrição */
    .detalhes-solicitacao__texto-descricao {
        width: 100%;
        padding: 0 2rem;
        margin-bottom: 1rem;
        line-height: 1.6;
        color: var(--color04);
        text-overflow: ellipsis;
        font-size: 1.7rem;
        overflow-wrap: break-word;
   text-align: center;
    }

    /* Estilo para imagens dentro do corpo da seção de detalhes */
    .detalhes-solicitacao__corpo img {
        align-items: center;
    }

    /* Estilo para imagens na seção de detalhes */
    .detalhes-solicitacao img {
        max-width: 30rem;
        max-height: 60rem;
```

```
        display: block;
        height: auto;
        object-fit: cover;
        border-radius: 0.5rem;
        margin-bottom: 1rem;
    }

    /* Estilos para o contêiner de botões */
    .container-btn {
        width: 100vw;
        height: auto;
        display: flex;
        align-items: center;
        gap: 1rem;
        justify-content: center;
    }

    /* Estilo base para botões */
    .botao {
        display: inline-block;
        width: 25rem;
        padding: 1.2rem;
        text-align: center;
        border-radius: 2rem;
        font-size: 1.6rem;
        font-weight: bold;
        text-decoration: none;
        transition: all 0.3s ease-in-out;
        cursor: pointer;
        margin: 0 auto;
        display: flex;
        align-items: center;
        justify-content: center;
    }

    /* Estilo específico para o botão de reencaminhar */
    .botao-reencaminhar {
        background-color: var(--color05);
        border: 2px solid var(--color05);
        color: var(--color01);
    }

    /* Estilo para o botão de reencaminhar ao passar o mouse */
    .botao-reencaminhar:hover {
        background-color: transparent;
        border: 2px solid var(--color05);
        color: var(--color05);
        box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
        transform: scale(1.00);
```

```
}

/* Estilo específico para o botão de excluir */
.botao-excluir {
    background-color: var(--colorRed);
    color: var(--color01);
    border: none;
}

/* Estilo para o botão de excluir ao passar o mouse */
.botao-excluir:hover {
    background-color: #a90f0f;
    border: none;
}

/* Estilo para o botão de cadastro */
.botao-cadastro {
    background-color: transparent;
    border: 2px solid var(--color03);
    color: var(--color03);
}

/* Estilo do botão de cadastro ao passar o mouse */
.botao-cadastro:hover {
    background-color: var(--color03);
    color: var(--color06);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.00);
}

/* Estilo para o contêiner de botões */
.container-botao {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: row;
    gap: 2rem;
}

/* Estilos para o modal */
.modal {
    display: none;
    position: fixed;
    z-index: 1;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: auto;
```

```
        background-color: rgb(0, 0, 0);
        background-color: rgba(0, 0, 0, 0.4);
    }

    /* Estilo para o conteúdo do modal */
    .modal__conteudo {
        background-color: #fefefe;
        margin: 15% auto;
        padding: 2rem;
        border: 1px solid #888;
        width: 80%;
    }

    .imagens_modal{
        list-style: none;
    }

    .imagens_modal img{
        width: 40%;
    }

    /* Estilo para o botão de fechar no modal */
    .modal__fechar {
        color: #aaa;
        float: right;
        font-size: 2.8rem;
        font-weight: bold;
    }

    /* Estilo para o botão de fechar ao passar o mouse */
    .modal__fechar:hover,
    .modal__fechar:focus {
        color: black;
        text-decoration: none;
        cursor: pointer;
    }

    /* Estilos adicionais para o conteúdo do modal */
    .modal__conteudo {
        background-color: #fff;
        border-radius: 0.8rem;
        box-shadow: 0 0.4rem 2rem rgba(0, 0, 0, 0.1);
        padding: 2rem;
    }

    /* Estilo para títulos h2 dentro do modal */
    .modal__conteudo h2 {
        font-family: 'Georgia', serif;
        font-size: 2.4rem;
```

```
        color: #333;
        margin-top: 1.5rem;
        margin-bottom: 1rem;
        border-bottom: 0.2rem solid #eaeaea;
        padding-bottom: 0.5rem;
    }

    /* Estilo para subtítulos h3 dentro do modal */
    .modal__conteudo h3 {
        font-family: 'Georgia', serif;
        font-size: 2rem;
        color: #444;
        margin-top: 1.5rem;
        margin-bottom: 0.5rem;
    }

    /* Estilo para listas não ordenadas dentro do modal */
    .modal__conteudo ul {
        list-style-type: disc;
        margin-left: 2rem;
        color: #555;
    }

    /* Estilo para parágrafos dentro do modal */
    .modal__conteudo p {
        font-size: 1.4rem;
        font-weight: bold;
    }

    /* Estilo para itens de listas dentro do modal */
    .modal__conteudo ul li {
        margin-bottom: 0.5rem;
        font-size: 1.4rem;
    }

    /* Estilo para separadores (hr) dentro do modal */
    .modal__conteudo hr {
        border: 1px solid #eaeaea;
        margin: 20px 0;
    }

    /* ---------------------- MEDIA ------------------------ */
    /* Responsividade para telas menores */
    @media (max-width: 700px) {
        /* Estilo para imagens no corpo da seção de detalhes */
        .detalhes-solicitacao__corpo img {
            align-items: center;
        }
```

```
/* Estilo para imagens na seção de detalhes */
.detalhes-solicitacao img {
    max-width: 30rem;
    height: auto;
    border-radius: 0.5rem;
    margin-bottom: 1rem;
}

/* Estilo para o título na seção de detalhes */
.detalhes-solicitacao__titulo {
    font-size: 1.5rem;
}

/* Estilo para a seção de detalhes */
.detalhes-solicitacao {
    max-width: 70rem;
}

/* Estilo para o contêiner de botões */
.container-botao {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    gap: 2rem;
}

/* Estilo para títulos na seção de detalhes */
.detalhes-solicitacao__titulo,
.detalhes-solicitacao__rotulo-descricao {
    font-size: 2rem;
}

/* Estilo para o autor e texto da descrição */
.detalhes-solicitacao__autor,
.detalhes-solicitacao__texto-descricao {
    font-size: 1.8rem;
}
.detalhes-solicitacao__texto-descricao {
    max-width: 40rem;
}

/* Estilo para o contêiner de botões */
.container-btn {
    width: 100vw;
    height: auto;
    display: flex;
    align-items: center;
    gap: 1rem;
```

```
            justify-content: center;
            flex-direction: column;
        }
    }

    /* Estilo para ocultar elementos no carregamento */
    #container-header,
    #detalhes-solicitacao {
        visibility: hidden;
    }
```

As linhas a seguir são do Javascript, ele adiciona os efeitos de transição das páginas, envia mensagens de sucesso e erro - de acordo com as ações do usuário - e possui a função do botão que retorna para as páginas anteriores.

```
    document.addEventListener('DOMContentLoaded', function() {
            // Inicializa o ScrollReveal e define a configuração
global
            window.sr = ScrollReveal({
                reset: true
            });

            // Aplica o efeito de revelação ao elemento com o ID
'container-header'
            sr.reveal('#container-header', {
                duration: 500,
                origin: 'top',
                distance: '50px'
            });

            // Aplica o efeito de revelação ao elemento com o ID
'conteudo-principal'
            sr.reveal('#detalhes-solicitacao', {
                duration: 800,
                origin: 'bottom',
                distance: '100px',
                delay: 200,
                easing: 'ease-in-out',
            });
        });

    // Função para voltar para a tela das solicitações
    function funcVoltar() {
            window.location.href = '/RF004'
    }
```

```
    // Função que retorna todos os blocos do SENAI no campo do formulário
    function deletarSolicitacao(id_solicitacao) {
            Swal.fire({
                    title: "Você tem certeza?",
                    text:    "Isso    deletará    a    solicitação permanentemente!",
                    icon: "warning",
                    showCancelButton: true,
                    confirmButtonColor: "var(--colorGreen)",
                    cancelButtonColor: "#d33",
                    confirmButtonText: "Deletar",
                    cancelButtonText: "Cancelar"
                  }).then((result) => {
                    if (result.isConfirmed) {
                            $.ajax({
                          url:                          `/deletar-solicitacao/${id_solicitacao}`,
                                    type: 'GET',
                                    success: function(){
                                            Swal.fire({
                                                    title: "Deletado!",
                                                    text:        "A solicitação foi deletada com sucesso",
                                                    icon: "success",
                                                    showConfirmButton: false
                                            });

                                            setTimeout(() => {
                                                    window.location.href = '/RF004';
                                            }, 2500);
                                    },
                                    error: function(){
                                            Swal.fire({
                                                    icon: "error",
                                                    title: "Oops...",
                                                    text: "Falha ao deletar a solicitação",
                                                    showConfirmButton: false,
                                                    timer: 3500
                                                  });
                                    }
                            });
```

```
                    }
                });
        }
```

O Python na tela de detalhamento de serviço cria a rota - um endereço virtual para a tela -, determina funções que obtém informações do banco de dados sobre a solicitação, executa funções de direcionamento da solicitação e função de excluir da mesma.

```
    # Criando rota que retorna as solicitações
    @app.route("/retorna-solicitacoes/<pesquisa>")
    def mostrar_solicitacoes(pesquisa): # Função que retorna as solicitações
        servico = Solicitacao()
        solicitacoes = servico.recebimento_solicitacoes(pesquisa)
        return jsonify(solicitacoes), 200

    # Criando rota para os detalhes da solicitação
    @app.route('/RF004A/<rowid>')
    def pg_ver_solicitacao(rowid): # Função que renderiza a tela de detalhes da solicitação
        if "usuario" in session:
            saibamais=Solicitacao()
            detalhes = saibamais.recebimento_solicitacao(id_solicitacao=rowid)

            print(detalhes)

            return render_template("RF004A-detlSolic.html", campo_sala = detalhes[2], campo_servico = detalhes[1], campo_nome_solicitante = detalhes[4], campo_funcao_solicitante = detalhes[5], campo_descricao = detalhes[3], campo_id_solicitacao = detalhes[0], campo_foto = detalhes[6])
        else:
            return redirect("/")

    # Criando rota para deletar a solicitação de serviço
    @app.route("/deletar-solicitacao/<id_solicitacao>")
    def deletarSolicitacao(id_solicitacao): # Função que permite que o administrador delete a solicitação de serviço
        solicitacao = Solicitacao()
        if solicitacao.deletar_solicitacao(id_solicitacao):
            return jsonify({'mensagem':'Cadastro OK'}), 200
        else:
            return {'mensagem':'ERRO'}, 500
```

```
    # Função que mostra os detalhes da solicitação de serviço
        def recebimento_solicitacao(self,id_solicitacao):
            myBD = Connection.conectar()
            mycursor = myBD.cursor()

            sql =(f"SELECT id_solicitacao, sv.nome, id_sala, descricao, f.nome, IF(fn.id_funcao IS NULL, 'Administrador', fn.nome), s.foto AS nome_funcao, s.foto FROM tb_solicitacoes s JOIN tb_funcionarios f ON f.CPF_funcionario = s.CPF_funcionario LEFT JOIN tb_funcoes fn ON fn.id_funcao = f.id_funcao JOIN tb_servicos sv ON sv.id_servico = s.id_servico WHERE s.id_solicitacao = {id_solicitacao};")
            mycursor.execute(sql)
            recebimento= mycursor.fetchone()

            return recebimento
```

```
        # Função que deleta a solicitação
        def deletar_solicitacao(self, id_solicitacao):
            try:
                myBD = Connection.conectar()
                mycursor = myBD.cursor()

                mycursor.execute(f"DELETE FROM tb_solicitacoes WHERE id_solicitacao =
{id_solicitacao}")

                myBD.commit()

                return True
            except:
                return False
```

Apêndice J – Tela de Encaminhamento da Solicitação

Este apêndice apresenta a tela utilizada pelo administrador para designar um colaborador responsável por executar um serviço, permitindo também a definição da prioridade da solicitação (baixa, média ou alta). O código HTML organiza a estrutura da página colocando um botão de navegação que possibilita retornar à tela anterior, uma área destinada à seleção do colaborador responsável, botões para escolha da prioridade do serviço e, por fim, um botão de envio que realiza o encaminhamento da solicitação ao sistema.

```
    <!DOCTYPE html>
    <html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <title>Designação de Funcionário</title>

        <!-- Link Arquivo CSS Global -->
        <link rel="stylesheet" href="/static/CSS/global.css">
        <!-- Link Arquivo CSS -->
        <link       rel="stylesheet"       href="/static/CSS/RF005-
encamSolic.css">
        <!-- Link Arquivo JS -->
        <script            defer            src="/static/JS/RF005-
encamSolic.js"></script>
        <!-- Link para os ICONS -->
```

```
        <script     src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
        <!-- link Arquivo JS função data -->
        <script defer src="/static/JS/funcao-data.js"></script>
        <script defer src="/static/JS/funcao-horas.js"></script>
        <script defer src="/static/JS/funcao-voltar.js"></script>
        <!-- Link para o Ajax -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <!-- link JS biblioteca SweetAlert -->
        <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
        <!-- Importa ScrollReveal -->
        <script src="https://unpkg.com/scrollreveal"></script>
        <!-- Link para o arquivo JS separado -->
        <script defer src="/static/JS/scrollReveal.js"></script>
        <!-- Link para o usúario receber notificações -->
        <script                defer                type="module"
src="/static/JS/notificacoes.js"></script>
        </head>
        <body>
            <nav>
                <button         class="container-header_btn-voltar"
onclick="funcVoltar()"><i          class="fa-solid         fa-arrow-
left"></i>Voltar</button>
            </nav>
            <main>
                <section class="container-main">
                        <div              class="form_select-user"
value="form_select-user"   id="form_select-user"   name="form_select-
user">

                        </div>

                        <div class="form_btn-emergencia">
                            <div                  class="form_btn-
emergencia__titulo">Prioridade:</div>
                            <div class="botoes">
                                <div   class="opcaoUrgencia"   data-
urgency="baixa"             id="baixa"             name="prioridade"
onclick="selecionarPrioridade('baixa')">Baixa Urgência</div>
                                <div   class="opcaoUrgencia"   data-
urgency="media"             id="media"             name="prioridade"
onclick="selecionarPrioridade('media')">Média Urgência</div>
                                <div   class="opcaoUrgencia"   data-
urgency="alta"              id="alta"              name="prioridade"
onclick="selecionarPrioridade('alta')">Alta Urgência</div>
                            </div>
```

```
                            <div class="container_btn-enviar">
                                <button          class="btn-enviar"
type="submit"
onclick="realizarEncaminhamento({{campo_id_solicitacao}})">Encaminha
r Serviço</button>
                            </div>
                        </div>

                </section>
            </main>
        </body>
    </html>
```

O código CSS define estilos para a página de encaminhamento da solicitação. Onde no cabeçalho é determinado a ocupar 10% da altura da tela, há um botão "voltar" estilizado. Uma seção principal permite a seleção de funcionários em uma área rolável, o código também inclui uma seção fixa na parte inferior da tela para selecionar níveis de urgência, com transições de cores, os botões são estilizados com cores vivas e transições suaves. Além disso, há suporte para responsividade, ajustando elementos para diferentes tamanhos de tela.

```
    /* Estilo da Topo da Tela (nome do Usuario e data */
    header{
      width: 100vw;
      height: 10vh;
      background-color: var(--color04);
      display: flex;
      align-items: center;
      justify-content: space-between;

    }

    .container-header{
      display: flex;
      align-items: center;
    }

    .container-header img{
        width: auto;
        height: 5rem;
        padding: 0.5rem;
        border-radius: 100%;
```

```
}

.container-header_user-cargo{
    background-color: var(--color05);
    color: var(--color01);
    margin-right: 10rem;
    text-align: center;
    padding: 0.2rem;
    border-radius: 5rem;
}

/* Estilo da data */

.data{
  margin: 1rem;
  display: flex;
  gap: 0.5rem;
  justify-content: center;
  align-items: center;
  font-size: 1.5rem;
}
.dia-semana{
    color: var(--color05);
    font-weight: 600;
}

/* Estilo do Botão Voltar */
nav{
  width: 100vw;
  height: 6vh;
  display: flex;
  align-items: center;
  margin-top: 1rem;
}

.container-header_btn-voltar{
  margin: 1rem;
  padding: 0.5rem;
  width: 10rem;
  border-color: var(--color02);
  border-radius: 5rem;
  background: none;
  font-weight: 700;
  transition-duration: 0.2s;
  color: var(--color02);
}

.container-header_btn-voltar i{
  margin-right: 0.5rem;
```

```
}

.container-header_btn-voltar:hover{
  border-color: none;
  background-color: var(--color02);
  color: var(--color01);
}

/* Estilo da Escolha de Funcionario */
.form_select-user{
  width: 100vw;
  height: 60vh;
  overflow-y: scroll;
  overflow-x: auto;
  position: relative;
}

/* estilo da barra de rolagem */
::-webkit-scrollbar {
  background-color: transparent;
}

.caixa-opcao {
  max-height: 10rem;
  padding: 1rem 6rem 1rem 1rem;
  margin: 1rem 2rem 1rem 2rem;
  cursor: pointer;
  display: flex;
  align-items: center;
  justify-content: space-between;
  color: var(--color01);
  background-color: var(--color06);
  border: none;
  border-radius: 5rem;
  box-shadow: 0px -1px 7px rgba(20,23,53,0.3);
  transition-duration: 0.2s;
  border: 2px solid transparent;

}


/* Transição do select */
.caixa-opcao.selected {
  border: 2px solid var(--color03);
  box-shadow: none;
  transform: scale(1.1);
}

.caixa-opcao img{
```

```
        width: 6rem;
        height: 6rem;
        padding: 0.5rem;
        border-radius: 100%;
        object-fit: cover;
    }

    .opcao_user{
      display: flex;
      justify-content: center;
      flex-direction: column;
      align-items: center;
      font-size: 1.5rem;
    }

    /* Estilo da Escolha da urgencia/emergencia */

    .form_btn-emergencia{
        width: 100vw;
        height: 33vh;
        position: fixed;
        bottom: 0;
        z-index: 1;
        background-color: var(--color06);
        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        color: var(--color01);
        align-content: center;

    }

    .form_btn-emergencia__titulo{
      margin-left: 2rem;
      font-size: 1.5rem;
    }

    .botoes{
        display: flex;
        width: 100vw;
        align-items: center;
        justify-content: space-around;
        margin-top: 2rem;

    }

    .opcaoUrgencia{
        padding: 1.4rem;
        border: solid 2px var(--color01);
        border-radius: 1rem;
```

```
        transition-duration: 0.2s;
        font-size: 1.1rem;
    }

    .opcaoUrgencia.selected{
      color: var(--color03);
      border: solid 2px var(--color03);

    }

    /* Estilo do botão 'Enviar Solicitação' */
    .container_btn-enviar{
      text-align: center;
      padding-left: 1rem;
    }

    .btn-enviar{
      background-color: var(--color03);
      color: var(--color01);
      width: 20rem;
      padding: 1.5rem;
      text-align: center;
      border: 2px solid var(--color03);
      border-radius: 50px;
      font-size: 1.6rem;
      font-weight: bold;
      cursor: pointer;
      margin-top: 1.5rem;
      transition-duration: 0.2s;

    }

    .btn-enviar:hover{
          background: none;
          border: 2px solid var(--color03);
          color: var(--color03);
    }
    .opcaoUrgencia {
      cursor: pointer;
      transition: background-color 0.3s;
    }

    #baixa:hover {
    color:   rgb(0, 255, 42);
    border: 0.2rem solid  rgb(0, 255, 42);
    }

    #media:hover {
      color: yellow;
```

```
    border: 0.2rem solid yellow;
  }

  #alta:hover {
    border: 0.2rem solid rgb(255, 27, 27);
    color:  rgb(255, 27, 27);
  }

  .opcaoUrgencia.baixa {
    /* Estilo da Topo da Tela (nome do Usuario e data */
    header{
      width: 100vw;
      height: 10vh;
      background-color: var(--color04);
      display: flex;
      align-items: center;
      justify-content: space-between;

    }

    .container-header{
      display: flex;
      align-items: center;
    }

    .container-header img{
        width: auto;
        height: 5rem;
        padding: 0.5rem;
        border-radius: 100%;
    }

    .container-header_user-cargo{
        background-color: var(--color05);
        color: var(--color01);
        margin-right: 10rem;
        text-align: center;
        padding: 0.2rem;
        border-radius: 5rem;
    }

    /* Estilo da data */

    .data{
      margin: 1rem;
      display: flex;
      gap: 0.5rem;
      justify-content: center;
      align-items: center;
```

```
    font-size: 1.5rem;
}
.dia-semana{
    color: var(--color05);
    font-weight: 600;
}

/* Estilo do Botão Voltar */
nav{
    width: 100vw;
    height: 6vh;
    display: flex;
    align-items: center;
    margin-top: 1rem;
}

.container-header_btn-voltar{
    margin: 1rem;
    padding: 0.5rem;
    width: 10rem;
    border-color: var(--color02);
    border-radius: 5rem;
    background: none;
    font-weight: 700;
    transition-duration: 0.2s;
    color: var(--color02);
}

.container-header_btn-voltar i{
    margin-right: 0.5rem;
}

.container-header_btn-voltar:hover{
    border-color: none;
    background-color: var(--color02);
    color: var(--color01);
}

/* Estilo da Escolha de Funcionario */
.form_select-user{
    width: 100vw;
    height: 60vh;
    overflow-y: scroll;
    overflow-x: auto;
    position: relative;
}

/* estilo da barra de rolagem */
::-webkit-scrollbar {
```

```
    background-color: transparent;
}

.caixa-opcao {
    max-height: 10rem;
    padding: 1rem 6rem 1rem 1rem;
    margin: 1rem 2rem 1rem 2rem;
    cursor: pointer;
    display: flex;
    align-items: center;
    justify-content: space-between;
    color: var(--color01);
    background-color: var(--color06);
    border: none;
    border-radius: 5rem;
    box-shadow: 0px -1px 7px rgba(20,23,53,0.3);
    transition-duration: 0.2s;
    border: 2px solid transparent;

}


/* Transição do select */
.caixa-opcao.selected {
    border: 2px solid var(--color03);
    box-shadow: none;
    transform: scale(1.1);
}

.caixa-opcao img{
    width: 6rem;
    height: 6rem;
    padding: 0.5rem;
    border-radius: 100%;
    object-fit: cover;
}

.opcao_user{
    display: flex;
    justify-content: center;
    flex-direction: column;
    align-items: center;
    font-size: 1.5rem;
}

/* Estilo da Escolha da urgencia/emergencia */

.form_btn-emergencia{
    width: 100vw;
```

```
        height: 33vh;
        position: fixed;
        bottom: 0;
        z-index: 1;
        background-color: var(--color06);
        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        color: var(--color01);
        align-content: center;

    }

    .form_btn-emergencia__titulo{
      margin-left: 2rem;
      font-size: 1.5rem;
    }

    .botoes{
        display: flex;
        width: 100vw;
        align-items: center;
        justify-content: space-around;
        margin-top: 2rem;

    }

    .opcaoUrgencia{
        padding: 1.4rem;
        border: solid 2px var(--color01);
        border-radius: 1rem;
        transition-duration: 0.2s;
        font-size: 1.1rem;
    }

    .opcaoUrgencia.selected{
      color: var(--color03);
      border: solid 2px var(--color03);

    }

    /* Estilo do botão 'Enviar Solicitação' */
    .container_btn-enviar{
      text-align: center;
      padding-left: 1rem;
    }

    .btn-enviar{
      background-color: var(--color03);
      color: var(--color01);
```

```
    width: 20rem;
    padding: 1.5rem;
    text-align: center;
    border: 2px solid var(--color03);
    border-radius: 50px;
    font-size: 1.6rem;
    font-weight: bold;
    cursor: pointer;
    margin-top: 1.5rem;
    transition-duration: 0.2s;

}

.btn-enviar:hover{
      background: none;
      border: 2px solid var(--color03);
      color: var(--color03);
}
.opcaoUrgencia {
    cursor: pointer;
    transition: background-color 0.3s;
}

#baixa:hover {
color:   rgb(0, 255, 42);
border: 0.2rem solid  rgb(0, 255, 42);
}

#media:hover {
    color: yellow;
    border: 0.2rem solid yellow;
}

#alta:hover {
    border: 0.2rem solid rgb(255, 27, 27);
    color:  rgb(255, 27, 27);
}

.opcaoUrgencia.baixa {
    color: rgb(0, 255, 42);
    border: solid 2px  rgb(0, 255, 42);
}

.opcaoUrgencia.media {
    color: yellow;
    border: solid 2px yellow;
}

.opcaoUrgencia.alta {
```

```
        color: rgb(255, 27, 27);
        border: solid 2px rgb(255, 27, 27);
      }

      /* ----------RESPONSIVIDADE----------- */
      @media (min-width:800px){
        .form_select-user{
          display: grid;
          grid-template-columns: 1fr 1fr;
          padding: 0rem 5rem;
          justify-content: space-between;
          align-items: center;
          align-content: space-evenly;

        }
        .opcaoUrgencia {
          padding: 2rem;
          font-size: 1.5rem;
        }

        .form_btn-emergencia__titulo {
          margin-left: 4rem;
        }
      }

      @media (min-width:1000px){
        .form_select-user{
          display: grid;
          grid-template-columns: 1fr 1fr;
          padding: 0rem 5rem;

        }

        .form_btn-emergencia__titulo {
          margin-left: 7rem;
        }
      }
      color: rgb(0, 255, 42);
      border: solid 2px  rgb(0, 255, 42);
    }

    .opcaoUrgencia.media {
      color: yellow;
      border: solid 2px yellow;
    }

    .opcaoUrgencia.alta {
      color: rgb(255, 27, 27);
      border: solid 2px rgb(255, 27, 27);
```

```
}

/* ----------RESPONSIVIDADE---------- */
@media (min-width:800px){
  .form_select-user{
    display: grid;
    grid-template-columns: 1fr 1fr;
    padding: 0rem 5rem;
    justify-content: space-between;
    align-items: center;
    align-content: space-evenly;

  }
  .opcaoUrgencia {
    padding: 2rem;
    font-size: 1.5rem;
  }

  .form_btn-emergencia__titulo {
    margin-left: 4rem;
  }
}

@media (min-width:1000px){
  .form_select-user{
    display: grid;
    grid-template-columns: 1fr 1fr;
    padding: 0rem 5rem;

  }

  .form_btn-emergencia__titulo {
    margin-left: 7rem;
  }
}
```

O código JavaScript implementa funcionalidades para gerenciar a seleção de funcionários e prioridades do encaminhamento de serviços. Ele utiliza o *ScrollReveal* para aplicar animações quando a página é carregada e faz requisições AJAX para buscar e exibir funcionários disponíveis. Cada funcionário é apresentado em cards, permitindo que o chefe de manutenção selecione um por vez. A prioridade do serviço é definida, com botões que destacam a opção escolhida e desmarcam as demais. Assim posteriormente elas são enviadas via POST para encaminhar o

serviço. Após o encaminhamento, o sistema busca um *token* do funcionário e envia uma notificação informando-o sobre o novo serviço.

```
      document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal
        window.sr = ScrollReveal({
            reset: true // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Aplica o efeito de revelação à parte de urgência
        sr.reveal('.form_btn-emergencia', {
            duration: 600,
            origin: 'bottom', // O elemento surge de baixo
            distance:  '50px',  //  Distância  que  o  elemento  vai
percorrer
            delay: 200 // Atraso antes de iniciar a animação
        });
      });

      // Definindo variáveis e selecionando elementos
      const  divFuncionarios  =  document.getElementById('form_select-
user');
      var funcionarioSelecionado;
      var prioridadeSelecionada;

      // Função que mostra todos os funcionários da manutenção
      function mostraFuncionarios(){
        $.ajax({
          url: '/RF005-retorna-funcionarios',
          type: 'GET',
          success: function(retorna_funcionarios){
              for (let x = 0; x < retorna_funcionarios.length; x++) {
                console.log(retorna_funcionarios[x])

                var div = document.createElement('div');

                div.className = 'caixa-opcao';
                div.id = `${retorna_funcionarios[x][1]}`
                div.setAttribute('onclick',
`selecionarFuncionario('${retorna_funcionarios[x][1]}');`);
                if (retorna_funcionarios[x][2]) {
                  div.innerHTML            =            `<figure><img
src="${retorna_funcionarios[x][2]}" alt="Foto de Perfil"></figure>
                                  <div
class="opcao_user">${retorna_funcionarios[x][0]}</div>`;
                }else{
```

```
                div.innerHTML           =           `<figure><img
src="https://st4.depositphotos.com/11574170/25191/v/450/depositphoto
s_251916955-stock-illustration-user-glyph-color-icon.jpg"   alt="Foto
de Perfil"></figure>
                <div
class="opcao_user">${retorna_funcionarios[x][0]}</div>`;
              }

              divFuncionarios.append(div);
            }
          },
          error: function(){
            Swal.fire({
              icon: "error",
              title: "Oops...",
              text: "Falha no Encaminhamento!",
              showConfirmButton: false,
              timer: 3500
            });
          }
      });
      }

      // Função para selecionar o funcionário
      function selecionarFuncionario(id_funcionario) {
        // Se já houver um funcionário selecionado, remove a classe
'selected' dele
        if (funcionarioSelecionado) {
          document.getElementById(`${funcionarioSelecionado}`).classL
ist.remove('selected');
        }

        // Se o funcionário clicado for diferente do atualmente
selecionado, adiciona a classe
        if (funcionarioSelecionado !== id_funcionario) {
          funcionarioSelecionado = id_funcionario;
          document.getElementById(`${id_funcionario}`).classList.add(
'selected');
        } else {
          // Se o funcionário clicado já estiver selecionado,
desmarca ele
          funcionarioSelecionado = '';
        }
        console.log(funcionarioSelecionado)
      }

      // Função para selecionar a prioridade
      function selecionarPrioridade(id_prioridade) {
        // Remover classes de todas as opções
```

```
        document.getElementById('baixa').classList.remove('selected',
'baixa');
        document.getElementById('media').classList.remove('selected',
'media');
        document.getElementById('alta').classList.remove('selected',
'alta');

        // Verifica se a prioridade já está selecionada
        if (prioridadeSelecionada === id_prioridade) {
          prioridadeSelecionada = '';
        } else {
          prioridadeSelecionada = id_prioridade;
          document.getElementById(prioridadeSelecionada).classList.ad
d('selected', prioridadeSelecionada);
        }
        console.log(prioridadeSelecionada);
      }

      // Função que realiza o encaminhamento do serviço
      function realizarEncaminhamento(id_solicitacao) {
        var dados = {
          id_solicitacao:id_solicitacao,
          CPF_funcionario:funcionarioSelecionado,
          prioridade:prioridadeSelecionada
        }

        console.log(dados);

        $.ajax({
          url: '/realizar-encaminhamento',
          type: 'POST',
          data: JSON.stringify(dados),
          contentType: 'application/json',
          success: function(){
            // Obter o token do administrador
            $.ajax({
              url: `/token-funcionario/${funcionarioSelecionado}`,
              type: 'GET',
              success: function(token){
                  var                tokenFuncionario                =
token['token_funcionario'][0];   //  Aqui  você  obtém  o  token  do
funcionario

                  // Agora que temos o token, podemos chamar a função
para enviar a notificação
                  enviarNotificacaoParaFuncionario(tokenFuncionario);
              },
              error: function(){
```

```
                swal("Oops!", "Erro no retorno do Token!",
"error");
            }
          });

        // Função para enviar a notificação para o administrador
        function enviarNotificacaoParaFuncionario(tokenFuncionario)
{
            fetch('/enviar-notificacao/encaminhamento', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    token: tokenFuncionario,
                    mensagem: "Um serviço foi encaminhado para
você!"  // Mensagem que você deseja enviar ao funcionario
                })
            })
            .then(response => response.json())
            .then(data => {
                console.log("Notificação enviada com sucesso:",
data);
            })
            .catch((error) => {
                console.error("Erro ao enviar notificação:",
error);
            });
        }
          Swal.fire({
            icon: "success",
            title: "Perfeito",
            text: "Serviço Encaminhado!",
            showConfirmButton: false,
            timer: 2500
          });
          setTimeout(() => {
            window.location.href = '/RF004';
          }, 2500);
        },
        error: function(){
          Swal.fire({
            icon: "error",
            title: "Oops...",
            text: "Falha no Encaminhamento!",
            showConfirmButton: false,
            timer: 3500
          });
        }
```

```
        })
    }

    // Executa a função que mostra os funcionários
    mostraFuncionarios();
```

O código do Python da tela de encaminhamento das solicitações define três rotas principais. A primeira rota, /RF005/<id_solicitacao> para renderizar a tela de encaminhamento de solicitações. A segunda rota, /realizar-encaminhamento, é responsável por receber dados via POST sobre a solicitação, ela cria um objeto da classe “Encaminhamento” e chama o método realizar_encaminhamento para processar e registrar o encaminhamento. Se o encaminhamento for bem-sucedido, retorna uma mensagem de sucesso, caso contrário, retorna um erro. A terceira rota, /RF005-retorna-funcionarios, retorna uma lista de funcionários disponíveis para a manutenção.

```
    # Criando rota para efetuar o encaminhamento
    @app.route("/realizar-encaminhamento", methods=["POST"])
    def realizar_encaminhamento(): # Função para realizar o encaminhamento
        dados = request.get_json()
        id_solicitacao = dados["id_solicitacao"]
        CPF_funcionario = dados["CPF_funcionario"]
        prioridade = dados["prioridade"]

        encaminhamento = Encaminhamento()

        if encaminhamento.realizar_encaminhamento(id_solicitacao, CPF_funcionario, prioridade):
            return jsonify({'mensagem':'Encaminhamento OK'}), 200
        else:
            return {'mensagem':'ERRO'}, 500

    # Criando rota para a tela que retorna funcionários
    @app.route("/RF005-retorna-funcionarios")
    def retorna_funcionarios(): # Função que retorna todos os funcionários da manutenção
        encaminhamento = Encaminhamento()
        retorna_funcionario = encaminhamento.mostra_funcionarios()
        print(retorna_funcionario)
        return jsonify(retorna_funcionario), 200
```

Apêndice K– Técnico de manutenção aceita o serviço

Esse apêndice contém a tela em que o técnico de manutenção aceita o serviço encaminhado pelo supervisor de manutenção. Nesta tela aparece os detalhes do serviço, para o técnico ficar ciente do serviço a ser realizado.

O HTML desta página insere todos os detalhes do serviço que foi encaminhado para o técnico de manutenção por meio da ferramenta jinja, que é um motor de templates que permite inserir lógicas python no HTML. Esses elementos são inseridos na parte principal do site(main).

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <!-- Definições básicas de meta tags: caracteres e
responsividade -->
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <!-- Importação de arquivos CSS: global e específico desta
página -->
    <link rel="stylesheet" href="/static/CSS/global.css">
    <link rel="stylesheet" href="/static/CSS/RF006A-
aceitaSolic.css">
    <!-- Link Arquivo JS -->
    <script defer src="/static/JS/RF006A -
aceitaSolic.js"></script>
    <!-- Biblioteca jQuery para manipulação de DOM e
requisições Ajax -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
    <!-- Biblioteca Font Awesome para uso de ícones na página -
->
    <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
    <!-- link JS biblioteca SweetAlert -->
    <script src="https://unpkg.com/scrollreveal"></script>
    <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <!-- Link para o usúario receber notificações -->
    <script defer type="module"
src="/static/JS/notificacoes.js"></script>

    <!-- Logo do Easy Request na guia-->
```

```
        <link      rel="shortcut      icon"      type="imagex/png"
href="/static/IMG/TASK request.svg">
        <title>Aceitar Solicitação</title>
    </head>
    <body>
        <!-- Navegação principal com botão de voltar -->
        <nav>
            <section    class="container-header"    id="container-
header">
                <div class="container-header_btn">
                    <button     class="container-header_btn-voltar"
onclick="funcVoltar()"><i          class="fa-solid          fa-arrow-
left"></i>Voltar</button>
                </div>
            </section>
        </nav>

        <!-- Conteúdo principal da página -->
        <main class="conteudo-principal">
            <section   class="detalhes-solicitacao"   id="detalhes-
solicitacao">
                <!-- Cabeçalho da solicitação com título e autor --
>
                <div class="detalhes-solicitacao__cabecalho">
                    <h1                             class="detalhes-
solicitacao__titulo">{{campo_sala}} - {{campo_servico}}</h1>
                    <p                              class="detalhes-
solicitacao__autor">{{campo_nome_solicitante}}                    -
{{campo_funcao_solicitante}}</p>
                    <p                              class="detalhes-
solicitacao__urgencia">{{campo_urgencia}}</p>
                </div>
                <!-- Linha divisora entre o cabeçalho e o corpo da
solicitação -->
                <hr class="detalhes-solicitacao__divisor">

                <!--  Corpo  da  solicitação  com  a  descrição  do
serviço e botão de encaminhamento -->
                <div class="detalhes-solicitacao__corpo">
                    <p         class="detalhes-solicitacao__rotulo-
descricao">Descrição</p>
                    <p          class="detalhes-solicitacao__texto-
descricao">
                        {{campo_descricao}}
                    </p>
                    {% if campo_foto == None %}
                    {%else%}
                    <img   src="{{campo_foto}}"   alt="Imagem   que
ilustra a solicitação">
```

```
                    {% endif %}
                    <!-- Botão para encaminhar a solicitação,
redirecionando para outra página -->
                    <button class="botao botao-cadastro"
onclick="iniciarServico({{campo_id_encaminhamento}})">ACEITAR</butto
n>
                    <!-- <a href="/iniciar-
servico/{{campo_id_encaminhamento}}"></a> -->
                </div>
            </section>
        </main>
    </body>
    </html>

                    <button class="botao botao-cadastro"
onclick="iniciarServico({{campo_id_encaminhamento}})">ACEITAR</butto
n>
                    <!-- <a href="/iniciar-
servico/{{campo_id_encaminhamento}}"></a> -->
                </div>
            </section>
        </main>
    </body>
    </html>
```

Na página em que o técnico de manutenção aceita o serviço o CSS é responsável pela estilização do cabeçalho e main, deixando a página intuitiva e de fácil visualização, e o @media está deixando tudo bem adaptado para várias telas.

```
    body{
        overflow-x: hidden;
    }
    a{
        text-decoration: none
    }
    nav{
        width: 100vw;
        height: 5vh;
        display: flex;
        justify-content: space-between;
    }

    .container-header_btn{
        display: flex;
```

```
        justify-content: center;
        align-items: center;
    }

    .container-header_btn-voltar{
        margin: 1rem;
        padding: 0.5rem;
        width: 10rem;
        border-color: var(--color02);
        border-radius: 5rem;
        background: none;
        font-weight: 700;

        transition-duration: 0.2s;
    }

    .container-header_btn-voltar i{
        margin-right: 0.5rem;
     }

     /* Mudança de Cor do botão ao passar o cursor por cima */
    .container-header_btn-voltar:hover{
        border-color: none;
        background-color: var(--color02);
        color: var(--color01);
    }



    /* Seção de detalhes da solicitação */
    .detalhes-solicitacao {
        width: 100vw;
        min-height: 75vh;
        max-height: 75vh;
        overflow-y: auto; /* Adiciona rolagem vertical */
        overflow-x: hidden; /* Esconde a rolagem horizontal */
        height: auto;
        background-color: var(--color06);
        margin-bottom: 0;
        position: absolute;
        bottom: 0;
        left: 0;

        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        padding: 1.5rem;

        display: flex;
        flex-direction: column;
```

```
        align-content: center;
        justify-content: space-evenly;
        box-shadow: -2px -1px 10px rgba(0, 0, 0, 0.5);
        padding-bottom: 3rem;
    }
    .detalhes-solicitacao__urgencia{
        margin: 0 auto;
        padding: 0.5rem 1rem;
        width: 30rem;
        border: 0.2rem solid var(--colorRed);
        color: var(--color01);
        font-size: 1.5rem;
        border-radius: 3rem;
        text-transform: capitalize;
    }
    .detalhes-solicitacao__cabecalho {
        text-align: center;
    }

    .detalhes-solicitacao__titulo {
        font-size: 4rem;
        color: var(--color05);
        margin-bottom: 0.5rem;
    }

    .detalhes-solicitacao__autor {
        font-size: 3.5rem;
        color: var(--color01);
        margin-bottom: 1rem;
        overflow-wrap: break-word;
    }

    .detalhes-solicitacao__divisor {
        border: none;
        border-top: 2px solid var(--color05);
        margin: 1.8rem 0;
    }

    .detalhes-solicitacao__corpo {
        padding: 0 3rem;
        font-size: 2.3rem;
        text-align: center;
        overflow-wrap: break-word;
        display: flex;
        flex-direction: column;
        align-items: center;

    }
```

```
.detalhes-solicitacao__rotulo-descricao {
    font-weight: bold;
    color: var(--color01);
    font-size: 3rem;
}

.detalhes-solicitacao__texto-descricao {
    width: 100%;
    padding: 0 2rem;
    margin-bottom: 1rem;
    line-height: 1.6;
    color: var(--color04);
    text-overflow: ellipsis;
    font-size: 1.7rem;
}

/* Estilos dos botões */
.botao {
    display: inline-block;
    width: 30rem;
    padding: 1.2rem;
    text-align: center;
    border-radius: 50px;
    font-size: 1.6rem;
    font-weight: bold;
    text-decoration: none;
    transition: all 0.3s ease-in-out;
    cursor: pointer;
    margin: 0 auto;
    display: flex;
    align-items: center;
    justify-content: center;
}
.botao-cadastro {
    background-color: transparent;
    border: 2px solid var(--color03);
    color: var(--color03);
}
.botao-cadastro{
    margin-top: 1rem;
}
/* Estilo do botão de cadastro ao passar o mouse */
.botao-cadastro:hover {
    background-color: var(--color03);
    color: var(--color06);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.05);
}
/* Estilo base para a imagem */
```

```
    img {
        max-width: 30rem;
        height: auto;
        display: block;
        height: auto;
        object-fit: cover;
        border-radius: 0.5rem;
        margin-bottom: 1rem;
    }
    /* Ajustes para telas menores que 300px */
    @media (max-width: 300px) {
        img {
            max-width: 90%;
        }
    }

    /* Ajustes específicos para telas ainda menores, como 280px */
    @media (max-width: 280px) {
        img {
            max-width: 85%; /* Reduz ainda mais a largura da imagem
*/
        }
    }

    /* Responsividade */
    @media (max-width: 700px) {
        .detalhes-solicitacao {
            font-size: 1.8rem;
        }
        .detalhes-solicitacao__autor{
            font-size: 1.8rem;
        }
        .detalhes-solicitacao__titulo {
            font-size: 1.8rem;
        }

        .detalhes-solicitacao__texto-descricao {
            font-size: 1.3rem;
            max-width: 60rem;
        }
        .detalhes-solicitacao{
            max-width: 70rem;
        }
    }
```

O Java Script que é responsável para deixar a página funcionando através das suas funções, essas funções são de carregamento da página automático

(setTimeout), os efeitos da página (sr.reveal) as mensagens de erro ou de sucesso (Swal.fire) e a função que faz o técnico de manutenção aceitar o serviço(function iniciarServico).

```
    document.addEventListener('DOMContentLoaded', function() {
      // Inicializa o ScrollReveal
      window.sr = ScrollReveal({
          reset: true // Efeito se repete sempre que o elemento
reaparece na tela
      });

      sr.reveal('.detalhes-solicitacao', {
        duration: 600,
        origin: 'bottom',
        distance: '50px',
        delay: 200
    });

      // Aplica o efeito de revelação ao container de baixo
      sr.reveal('.container-header', {
          duration: 600,
          origin: 'top',
          distance: '50px',
          delay: 200
      });
    });

    // Função que permite que o usuário inicie o serviço
    function iniciarServico(id_encaminhamento) {
        $.ajax({
          url: `/iniciar-servico/${id_encaminhamento}`,
          type: 'GET',
          success: function(){
            Swal.fire({
                position: "center",
                icon: "success",
                title: "Serviço Iniciado!",
                showConfirmButton: false,
                timer: 1500
              });

              setTimeout(() => {
                window.location.href = '/RF006';
            }, 1500);
          },
          error: function(mensagem){
            console.log(mensagem)
            Swal.fire({
```

```
                icon: "error",
                title: "Oops...",
                text: mensagem['responseText'],
                showConfirmButton: false,
                timer: 3500
              });
            }
          })
      }

      // Função para voltar para a tela inicial do funcionário da manutenção
      function funcVoltar() {
        window.location.href = '/RF006'
      }
```

O Python que é responsável pela conexão do banco de dados com o sistema, ele teve um papel crucial pois ele pegou os valores que estavam no banco de dados com o jsonify, para colocar os valores na página.

```
      def aceitar_encaminhamento(self, id_encaminhamento, cpf_funcionario):
              myBD = Connection.conectar()

              mycursor = myBD.cursor()

              self.cpf_funcionario = cpf_funcionario
              data_inicio_servico = datetime.now().strftime('%Y-%m-%d %H:%M:%S')

              mycursor.execute(f"SELECT status FROM tb_encaminhamentos WHERE CPF_funcionario = '{cpf_funcionario}' AND status = 'fazendo'")

              retorno = mycursor.fetchone()

              print(retorno)

              if retorno is None:
                  mycursor.execute(f"UPDATE tb_encaminhamentos SET status = 'fazendo', data_inicio_servico = '{data_inicio_servico}' WHERE id_encaminhamento = {id_encaminhamento};")
                  myBD.commit()
                  return True
```

```
            elif retorno[0] == "fazendo":
                return "Você só pode fazer um serviço por vez!"
```

```
    @app.route("/iniciar-servico/<id_encaminhamento>")
    def iniciar_servico(id_encaminhamento): # Função que inicia o
serviço
        encaminhamento = Encaminhamento()

        cpf_funcionario = session["usuario"]["CPF"]

        retorno                                                 =
encaminhamento.aceitar_encaminhamento(id_encaminhamento,
cpf_funcionario)

        if retorno == 'Você só pode fazer um serviço por vez!':
            return 'Você só pode fazer um serviço por vez!', 500
        elif retorno:
            return jsonify({'mensagem':'Encaminhamento OK'}), 200
        else:
            return 'Erro ao iniciar serviço!', 500
```

Apêndice L – Técnico de manutenção detalhamento do serviço

Este apêndice abrange os detalhes do serviço que foram enviados para o técnico de manutenção pelo supervisor de manutenção - administrador -. Além de possuir um botão que finaliza e redireciona para a tela de finalização do serviço.

O html nesta página estrutura o título, o texto com detalhe do serviço, imagem e botão de finalização.

```
    <!DOCTYPE html>
    <html lang="pt-br">
    <head>
        <!--  Definições  básicas  de  meta  tags:  caracteres  e
responsividade -->
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <!-- Importação de arquivos CSS: global e específico desta
página -->
        <link rel="stylesheet" href="/static/CSS/global.css">
        <link      rel="stylesheet"      href="/static/CSS/RF006B-
detalheSolic.css">
```

```
        <!-- Link Arquivo JS -->
        <script            defer            src="/static/JS/RF006B-
detalheSoclic.js"></script>
        <script defer src="/static/JS/funcao-voltar.js"></script>
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <script     src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
        <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
        <script src="https://unpkg.com/scrollreveal"></script>
        <script defer src="/static/JS/scrollReveal.js"></script>
        <!-- Link para o usuário receber notificações -->
        <script                defer                type="module"
src="/static/JS/notificacoes.js"></script>

        <!-- Logo do Easy Request na guia-->
        <link      rel="shortcut      icon"      type="imagex/png"
href="/static/IMG/TASK request.svg">

        <title>Detalhes da Solicitação</title>
    </head>
    <body>
        <nav>
            <section class="container-header">
                <div class="container-header_btn">
                    <button     class="container-header_btn-voltar"
onclick="funcVoltar()"><i          class="fa-solid          fa-arrow-
left"></i>Voltar</button>
                </div>
            </section>
        </nav>

        <main class="conteudo-principal">
            <section class="detalhes-solicitacao">
                <div class="detalhes-solicitacao__cabecalho">
                    <h1                          class="detalhes-
solicitacao__titulo">{{campo_sala}} - {{campo_servico}}</h1>
                    <p                           class="detalhes-
solicitacao__autor">{{campo_nome_solicitante}}                    -
{{campo_funcao_solicitante}}</p>
                </div>
                <hr class="detalhes-solicitacao__divisor">
                <div class="detalhes-solicitacao__corpo">
                    <p         class="detalhes-solicitacao__rotulo-
descricao">Descrição</p>
                    <p          class="detalhes-solicitacao__texto-
descricao">
```

```
                        {{campo_descricao}}
                    </p>
                    {% if campo_foto == None %}
                    {%else%}
                    <img   src="{{campo_foto}}"   alt="Imagem   que
ilustra a solicitação">
                    {% endif %}
                    <!--  Botão  para  encaminhar  a  solicitação,
redirecionando para outra página -->
                    <button      class="botao      botao-cadastro"
onclick="finalizarServico({{campo_id_encaminhamento}})">Finalizar</b
utton>
                </div>
            </section>
        </main>
    </body>
    </html>
```

A estilização - CSS - da tela de detalhamento do técnico de manutenção determinou o tamanho e cores dos elementos da página - título, botões, imagem e textos. Ademais, o CSS permite que a tela seja responsiva em diversos dispositivos.

```
    body{
        overflow-x: hidden;
    }
    a{
        text-decoration: none
    }
    nav{
        width: 100vw;
        height: 5vh;
        display: flex;
        justify-content: space-between;
    }

    .container-header_btn{
        display: flex;
        justify-content: center;
        align-items: center;
    }

    .container-header_btn-voltar{
        margin: 1rem;
        padding: 0.5rem;
```

```
        width: 10rem;
        border-color: var(--color02);
        border-radius: 5rem;
        background: none;
        font-weight: 700;

        transition-duration: 0.2s;
    }

    .container-header_btn-voltar i{
        margin-right: 0.5rem;
     }

     /* Mudança de Cor do botão ao passar o cursor por cima */
    .container-header_btn-voltar:hover{
        border-color: none;
        background-color: var(--color02);
        color: var(--color01);
    }



    /* Seção de detalhes da solicitação */
    .detalhes-solicitacao {
        width: 100vw;
        min-height: 75vh;
        max-height: 75vh;
        overflow-y: auto; /* Adiciona rolagem vertical */
        overflow-x: hidden; /* Esconde a rolagem horizontal */
        height: auto;
        background-color: var(--color06);
        margin-bottom: 0;
        position: absolute;
        bottom: 0;
        left: 0;

        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        padding: 1.5rem;

        display: flex;
        flex-direction: column;
        align-content: center;
        justify-content: space-evenly;
        box-shadow: -2px -1px 10px rgba(0, 0, 0, 0.5);
        padding-bottom: 3rem;
    }
    /* estilo da barra de rolagem */
    ::-webkit-scrollbar {
```

```
        background-color: transparent;
    }

    .detalhes-solicitacao__urgencia{
        margin: 0 auto;
        padding: 0.5rem 1rem;
        width: 30rem;
        border: 0.2rem solid var(--colorRed);
        color: var(--color01);
        font-size: 1.5rem;
        border-radius: 3rem;

    }
    .detalhes-solicitacao__cabecalho {
        text-align: center;
    }

    .detalhes-solicitacao__titulo {
        font-size: 4rem;
        color: var(--color05);
        margin-bottom: 0.5rem;
    }

    .detalhes-solicitacao__autor {
        font-size: 3.5rem;
        color: var(--color01);
        margin-bottom: 1rem;
        overflow-wrap: break-word;
    }

    .detalhes-solicitacao__divisor {
        border: none;
        border-top: 2px solid var(--color05);
        margin: 1.8rem 0;
    }

    .detalhes-solicitacao__corpo {
        padding: 0 3rem;
        font-size: 2.3rem;
        text-align: center;
        overflow-wrap: break-word;
        display: flex;
        flex-direction: column;
        align-items: center;

    }

    .detalhes-solicitacao__rotulo-descricao {
        font-weight: bold;
```

```
        color: var(--color01);
        font-size: 3rem;
    }

    .detalhes-solicitacao__texto-descricao {
        margin-bottom: 1rem;
        line-height: 1.6;
        color: var(--color04);
        text-overflow: ellipsis;
        font-size: 1.7rem;
        width: 100%;
        padding: 0 2rem;
    }

    /* Estilos dos botões */
    .botao {
        display: inline-block;
        width: 30rem;
        padding: 1.2rem;
        text-align: center;
        border-radius: 50px;
        font-size: 1.6rem;
        font-weight: bold;
        text-decoration: none;
        transition: all 0.3s ease-in-out;
        cursor: pointer;
        margin: 0 auto;
        display: flex;
        align-items: center;
        justify-content: center;
    }
    .botao-cadastro {
        background-color: transparent;
        border: 2px solid var(--color03);
        color: var(--color03);
    }

    /* Estilo do botão de cadastro ao passar o mouse */
    .botao-cadastro:hover {
        background-color: var(--color03);
        color: var(--color06);
        box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
        transform: scale(1.05);
    }

    .botao-cadastro{
        margin-top: 1rem;
    }
    /* Estilo base para a imagem */
```

```
    img {
        max-width: 40vw; /* Garante que a imagem não exceda a
largura do contêiner */
        height: auto; /* Mantém a proporção da imagem */
        object-fit: cover; /* Ajusta a imagem para cobrir o espaço
disponível */
        border-radius: 0.5rem;
        margin-bottom: 1rem;
        display: block;
        margin: 0 auto;
    }

    /* Ajustes para telas menores que 300px */
    @media (max-width: 300px) {
        img {
            max-width: 90%; /* Reduz a largura da imagem em telas
menores */
        }
    }

    /* Ajustes específicos para telas ainda menores, como 280px */
    @media (max-width: 280px) {
        img {
            max-width: 85%; /* Reduz ainda mais a largura da imagem
*/
        }
    }

    /* Responsividade */
    @media (max-width: 700px) {
        .detalhes-solicitacao {
            font-size: 1.8rem;
        }
        .detalhes-solicitacao__autor{
            font-size: 1.8rem;
        }
        .detalhes-solicitacao__titulo {
            font-size: 1.8rem;
        }

        .detalhes-solicitacao__texto-descricao {
            font-size: 1.3rem;
            max-width: 60rem;
        }
        .detalhes-solicitacao{
            max-width: 70rem;
        }
    }
```

Nesta parte, onde o código Javascript se insere, adiciona na tela de detalhamento do técnico de manutenção efeitos de transição e a execução da função do botão de finalizar o serviço - redirecionando para a tela de finalização do serviço -.

```
    document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal
        window.sr = ScrollReveal({
            reset: true // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Aplica o efeito de revelação à parte da solicitação
        sr.reveal('.detalhes-solicitacao', {
            duration: 600,
            origin: 'bottom', // A seção surge de baixo
            distance:  '50px',  //  Distância  que  a  seção  vai
percorrer
            delay: 200 // Atraso antes de iniciar a animação
        });
    });

    // Função para ir para a tela de finalizar serviço
    function finalizarServico(id_encaminhamento) {
        console.log(id_encaminhamento);
        window.location.href = `/RF007/${id_encaminhamento}`;
    }
```

Por fim, no código python desta tela é executada a rota da página e a conexão do banco de dados para a exibição dos dados do serviço.

```
    # Criando a rota para a tela em que o funcionário pode ver os
detalhes do serviço
    @app.route("/RF006B/<id_solicitacao>/<id_encaminhamento>")
    def pg_detalhes_solicitacao(id_solicitacao, id_encaminhamento):
# Função que renderiza a tela de detalhes de serviço
        if "usuario" in session:
            solicitacao = Solicitacao()
            detalhes                                          =
solicitacao.recebimento_solicitacao(id_solicitacao)

            print(detalhes)
```

```
            return      render_template("RF006B-detalheSolic.html",
campo_sala    =    detalhes[2],    campo_servico    =    detalhes[1],
campo_nome_solicitante  =  detalhes[4],  campo_funcao_solicitante  =
detalhes[5],  campo_descricao  =  detalhes[3],  campo_id_solicitacao  =
detalhes[0],  campo_foto  =  detalhes[6],  campo_id_encaminhamento  =
id_encaminhamento)
        else:
            return redirect("/")
```

Apêndice M – Menu de confirmação do Serviço

O código HTML em questão define a estrutura de uma página destinada à confirmação de finalização de serviço. O cabeçalho da página contém um botão de "Voltar", que permite ao usuário retornar à página anterior. No corpo da página é disposto um formulário com opções de resposta para a conclusão do serviço, como "Não foi possível realizar a tarefa", "Foi realizado, mas necessita de outros cuidados" e "Tarefa foi totalmente realizada". Também há um campo de texto para adicionar informações adicionais, caso necessário, além de um campo para o envio de uma foto opcional. O formulário é finalizado com um botão para submeter os dados, que envia as informações para o servidor ao ser clicado. O código utiliza JavaScript para validação do envio do formulário e jQuery para facilitar a manipulação de dados e eventos na página.

```
    <!DOCTYPE html>
    <html lang="pt-br">

    <head>
        <!--  Definindo  a  codificação  de  caracteres  e  a
responsividade -->
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">

        <!-- Link para o CSS Global -->
```

```
        <link rel="stylesheet" href="/static/CSS/global.css">
        <!-- Link para o CSS específico desta página -->
        <link rel="stylesheet" href="/static/CSS/RF007-
manutConfirm.css">
        <!-- Scripts JavaScript: Carregamento dos arquivos de
script da página -->
        <script src="/static/JS/RF007-manutConfirm.js"></script>
        <!-- Biblioteca de ícones Font Awesome -->
        <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
        <!-- link JS biblioteca SweetAlert -->
        <script src="https://unpkg.com/scrollreveal"></script>
        <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
        <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>
        <!-- Link para o usúario receber notificações -->
        <script defer type="module"
src="/static/JS/notificacoes.js"></script>

        <!-- Logo do Easy Request na guia-->
        <link rel="shortcut icon" type="imagex/png"
href="/static/IMG/TASK request.svg">
        <!-- Título da página -->
        <title>Finalizar Serviço</title>
    </head>
    <body>
        <!-- Cabeçalho com informações do usuário e botão de voltar
-->
        <header class="cabecalho" id="cabecalho">
            <div class="cabecalho__botao-container">
                <button class="cabecalho__botao-voltar"
onclick="funcVoltar()">
                    <i class="fa-solid fa-arrow-left"></i>Voltar
                </button>
            </div>

        </header>

        <!-- Conteúdo principal da página -->
        <main class="principal" id="principal">
            <div class="principal__container"
id="principal__container">
                <!-- Título da seção principal -->
                <p class="principal__titulo">Confirmar finalização
de serviço</p>
```

```
                <!-- Formulário de finalização de tarefa -->

                <form  class="formulario-finalizar"  action="/RF007"
method="post"      enctype="multipart/form-data"      onsubmit="return
enviarNotificacao()">
                    <input    type="hidden"    name="id_solicitacao"
value="{{campo_id_encaminhamento}}">
                    <!-- Opções de finalização da tarefa -->
                    <div class="formulario-finalizar__opcoes">
                        <p                          class="formulario-
finalizar__titulo">Escolha uma das opções:</p>

                        <label class="formulario-finalizar__opcao">
                            <input     type="radio"     name="opcao"
value="pendente">
                            <span                 class="formulario-
finalizar__texto ">Não foi possível realizar a tarefa</span>
                        </label>
                        <label class="formulario-finalizar__opcao">
                            <input     type="radio"     name="opcao"
value="realizado">
                            <span                 class="formulario-
finalizar__texto   ">Foi   realizado,   mas   necessita   de   outros
cuidados</span>
                        </label>
                        <label class="formulario-finalizar__opcao">
                            <input     type="radio"     name="opcao"
value="completo">
                            <span                 class="formulario-
finalizar__texto ">Tarefa foi totalmente realizada</span>
                        </label>
                    </div>

                    <!-- Caixa de texto para informações adicionais
-->
                    <div class="formulario-finalizar__caixa-texto">
                        <p                          class="formulario-
finalizar__titulo">Inserir informações adicionais (caso houver):</p>
                        <textarea  name="adendo"  class="formulario-
finalizar__comentarios" placeholder="Digite aqui"></textarea>
                        <div class="form_input-arquivo">
                            <label                    for="foto">Foto
(Opcional):</label>
                            <input      type="file"      name="foto"
id="foto">
                        </div>
                        <!-- Botão para finalizar a tarefa -->
```

```
                        <button   type="submit"   class="formulario-
finalizar__botao">Finalizar tarefa</button>
                    </div>
                </form>
            </div>
        </main>
    </body>
    </html>
```

O código CSS da tela do menu de confirmação utiliza o Flexbox para organizar os elementos, como o cabeçalho e a barra de navegação. Ele inclui efeitos de hover e transições suaves para botões e caixas de seleção, As cores e sombras são ajustáveis via variáveis. A responsividade é garantida por media queries, adaptando o layout para diferentes tamanhos de tela e proporcionando uma boa experiência em dispositivos móveis e desktop.

```
    /* Estilo para o body */
    body {
        overflow-x: hidden;
        display: flex;
        flex-direction: column;
        justify-content: flex-start;
        height: 100vh;
        margin: 0;
    }

    /* Cabeçalho */
    .cabecalho {
        width: 100%;
    }

    .cabecalho__botao-container {
        margin: 0.5rem 0 0 0.5rem;
        display: flex;
        justify-content: flex-start;
        align-items: center;
        height: 11vh;
    }

    .cabecalho__botao-voltar {
        margin: 1rem;
        padding: 0.5rem;
        width: 10rem;
        border: 2px solid var(--color02);
        border-radius: 5rem;
```

```
        background: none;
        font-weight: 700;
        transition-duration: 0.2s;
        display: flex;
        justify-content: center;
        align-items: center;
    }

    .cabecalho__botao-voltar i {
        margin-right: 0.5rem;
    }

    .cabecalho__botao-voltar:hover {
        background-color: var(--color02);
        color: var(--color01);
    }

    /* Container principal - Fixo na parte inferior */
    .principal__container {
        min-height: 75vh;
        max-height: 75vh;
        overflow-y: auto; /* Adiciona rolagem vertical */
        overflow-x: hidden; /* Esconde a rolagem horizontal */
        background-color: var(--color06);
        margin-bottom: 0;
        padding: 1.5rem;
        display: flex;
        flex-direction: column;
        justify-content: center; /* Alinha tudo ao centro */
        align-items: center; /* Centraliza todos os elementos */
        border-top-left-radius: 2rem;
        border-top-right-radius: 2rem;
        box-shadow: -7px 7px 16px black;
        position: fixed; /* Fixa o container na tela */
        bottom: 0; /* Posiciona o container na parte inferior */
        left: 0; /* Alinha o container à esquerda */
        right: 0; /* Alinha o container à direita */
        z-index: 100; /* Garante que o container fique acima de
outros elementos */
        overflow-y: auto; /* Permite rolagem vertical caso o
conteúdo ultrapasse a altura disponível */
        padding-top: 2rem;
    }


    /* Título principal */
    .principal__titulo {
        color: var(--color01);
        font-size: 2.5rem;
```

```
        text-align: center;
        margin-bottom: 1rem;
        padding: 1rem 3rem 0 3rem;
    }

    /* Estilo do Formulário */
    .formulario-finalizar {
        width: 100%;
        padding: 1.5rem;
        display: flex;
        flex-direction: column;  /* Itens em coluna */
        gap: 1.5rem;
        align-items: center; /* Centraliza os itens */
    }

    /* Título das opções do formulário */
    .formulario-finalizar__titulo {
        color: var(--color01);
        font-size: 1.8rem;
        margin-bottom: 1rem;
        text-align: center;
    }

    /* Estilo das opções de rádio */
    .formulario-finalizar__opcao {
        display: flex;
        align-items: center;
        gap: 0.7rem;
        padding: 1rem 0;
    }

    .formulario-finalizar__input {
        accent-color: var(--color04);
    }

    .formulario-finalizar__texto {
        color: var(--color01);
        font-size: 1.6rem;
        padding: 0.5rem 0 0.5rem 0;
    }

    /* Campo de comentários */
    .formulario-finalizar__comentarios {
        width: 100%;
        height: 9rem;
        padding: 1rem;
        border-radius: 0.5rem;
        border: 1px solid var(--color01);
        font-size: 1.3rem;
```

```
        color: var(--color01);
        background-color: var(--color06);
        resize: none;
        margin: 0 0 2rem 0;
    }

    /* Input de arquivo */
    .form_input-arquivo {
        margin-bottom: 2rem;
        color: var(--color01);
        font-size: 1.5rem;
        display: flex;
        flex-direction: column;
        gap: 1rem;
        align-items: center; /* Centraliza o input de arquivo */
    }

    /* Botão de Finalizar */
    .formulario-finalizar__botao {
        width: 100%;
        padding: 1rem;
        background-color: var(--color05);
        color: var(--color01);
        font-size: 1.2rem;
        font-weight: bold;
        border-radius: 2rem;
        border: none;
        cursor: pointer;
        transition: background-color 0.3s cubic-bezier(0, -0.02,
0.64, 0.69);
        text-align: center; /* Centraliza o texto dentro do botão
*/
    }

    .formulario-finalizar__botao:hover {
        background-color: var(--color03);
    }

    /* Estilo para o input do tipo "radio" */
    input[type="radio"] {
        width: 2.5rem;
        height: 2.5rem;
        appearance: none;
        -webkit-appearance: none;
        -moz-appearance: none;
        border: 2px solid var(--color01);
        border-radius: 50%;
        outline: none;
        cursor: pointer;
```

```
}

input[type="radio"]:checked {
    background-color: var(--color05);
    border: 2px solid var(--color05);
}

/* Ajustes de Responsividade */
@media (max-width: 400px) {

    .principal__titulo{
        font-size: 2.3rem;
    }
    .formulario-finalizar__titulo{
        font-size: 2rem;
    }
    .formulario-finalizar__texto {
        font-size: 1.6rem;
    }
  }

  @media (min-width: 750px) {
    .principal__container {
        padding-top: 0 ;
    }
    .principal__titulo{
        font-size: 3.5rem;
    }
    .formulario-finalizar__titulo{
        font-size: 2.5rem;
    }
    .formulario-finalizar__texto {
        font-size: 2rem;
    }
  }
```

O código JavaScript da tela do menu de confirmação utilizá a biblioteca ScrollReveal para criar animações de entrada para os elementos quando a tela é iniciada. A função, funcVoltar(), redireciona o usuário para a tela inicial do funcionário da manutenção. A segunda, enviarNotificacao(), utiliza uma requisição AJAX para obter o token de administrador e, em seguida, envia uma notificação sobre a finalização de um serviço. O código lida com chamadas assíncronas e trata erros ao longo do processo.

```
    document.addEventListener('DOMContentLoaded', function() {
        // Inicializa o ScrollReveal e define a configuração global
        window.sr = ScrollReveal({
            reset: true  // Efeito se repete sempre que o elemento
reaparece na tela
        });

        // Aplica o efeito de revelação ao elemento com o ID
'container-header'
        sr.reveal('#cabecalho', {
            duration: 500,  // Duração da animação em milissegundos
            origin: 'top',   // Elemento surge do topo
            distance: '50px' // Distância percorrida pelo elemento
        });

        // Aplica o efeito de revelação ao elemento com o ID
'conteudo-principal'
        sr.reveal('#principal__container', {
            duration: 800,  // Duração da animação (mais lenta)
            origin: 'bottom',  // Elemento surge de baixo
            distance:  '100px',  // Distância que o elemento vai
percorrer
            delay: 200,  // Atraso antes de iniciar a animação
            easing:  'ease-in-out',   //  Tipo  de  suavização  do
movimento
        });
    });

    // Função para ir para a tela inicial do funcionário da
manutenção
    function funcVoltar() {
        window.location.href = '/RF006';
    }

    // Função para enviar notificação quando o serviço for
finalizado
    function enviarNotificacao() {
        // Obter o token do administrador
        $.ajax({
            url: '/token-administrador',
            type: 'GET',
            success: function(token){
                if(token['permissao'] == 'administrador'){
                    return;
                }else{
                    var            tokenAdministrador            =
token['token_administrador'][0];   // Aqui você obtém o token do
administrador
                }
```

```
                // Agora que temos o token, podemos chamar a função
para enviar a notificação
                enviarNotificacaoParaAdministrador(tokenAdministrad
or);
            },
            error: function(){
                swal("Oops!",   "Erro   no   retorno   do   Token!",
"error");
            }
        });

        // Função para enviar a notificação para o administrador
        function
enviarNotificacaoParaAdministrador(tokenAdministrador) {
            fetch('/enviar-notificacao/finalizacao', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    token: tokenAdministrador,
                    mensagem: "Um novo serviço foi finalizado!"  //
Mensagem que você deseja enviar ao administrador
                })
            })
            .then(response => response.json())
            .then(data => {
                console.log("Notificação   enviada   com   sucesso:",
data);
            })
            .catch((error) => {
                console.error("Erro   ao   enviar   notificação:",
error);
            });
        }
    }
```

O codigo Python define duas rotas no Flask para gerenciar a finalização de serviços por funcionários da manutenção. A primeira rota (/RF007/<id_encaminhamento>) renderiza a tela de finalização, recuperando dados do usuário logado (nome e foto) e passando-os para o HTML. A segunda rota (/RF007), utilizando o método POST, coleta informações de um formulário (ID da

solicitação, adendo, opção e foto), e chama um método para finalizar o serviço no banco de dados.

```
    # Criando a rota para tela de finalização de serviços (Funcionário da Manutenção)
    @app.route("/RF007/<id_encaminhamento>", methods=["GET"])
    def pg_manutencao_confirmacao_get(id_encaminhamento): # Função que renderiza a tela de finalização de serviços (Funcionário da Manutenção)
        if "usuario" in session:
            foto = session["usuario"]["foto"]
            nome_completo = session["usuario"]["nome"]

            # Dividindo o nome em partes
            partes_nome = nome_completo.split()

            # Verificando se o nome tem mais de uma parte
            if len(partes_nome) > 1:
                primeiro_ultimo_nome = f"{partes_nome[0]} {partes_nome[-1]}"
            else:
                # Caso o nome tenha apenas uma parte, retorna somente essa parte
                primeiro_ultimo_nome = partes_nome[0]

            return render_template("RF007-manutConfirm.html", campo_id_encaminhamento = id_encaminhamento, campo_foto = foto, campo_nome = primeiro_ultimo_nome)
        else:
            return redirect("/")

    # Criando rota que finaliza o serviço(Funcionário da Manutenção)
    @app.route("/RF007", methods=["POST"])
    def pg_manutencao_confirmacao_post(): # Função que finaliza o serviço, enviado as informações para o banco de dados(Funcionário da Manutenção)
        id_encaminhamento = request.form.get("id_solicitacao")
        adendo = request.form.get("adendo")
        opcao = request.form.get("opcao")
        foto = request.files.get("foto")

        print(foto)

        if foto:
            link_arquivo_foto = upload_file(foto)
        else:
            link_arquivo_foto = None
```

```
        encaminhamento = Encaminhamento()
        encaminhamento.finalizacao_encaminhamento(id_encaminhamento
, adendo, opcao, link_arquivo_foto)

        return render_template("RF007-manutConfirm.html")
```

Apêndice N – Administrador finaliza o serviço

Neste apêndice o administrador - após o técnico de manutenção finalizar o serviço - a solicitação é analisada e o supervisor de manutenção pode reencaminhar o serviço - caso não tenha sido efetuado corretamente - ou finalizar. Após isto é gerado um modal - elemento sobreposto na tela - contendo todas as informações do serviço realizado.

O html estrutura o local do botão que retorna à página anterior, posiciona o título, filtro e os elementos do serviço dentro do card.- local de informações de serviço-.

```
<!DOCTYPE html>

<html lang="pt-br">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-
scale=1.0">

    <link rel="stylesheet" href="/static/CSS/global.css">

    <link rel="stylesheet" href="/static/CSS/RF004A-detlSolic.css">

    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>

    <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>
```

```
    <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>

    <script src="https://unpkg.com/scrollreveal"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/dompurify/2.3.10/purify.
min.js"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/html2c
anvas.min.js"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.mi
n.js"></script>

    <script defer src="/static/JS/RF009-
relatorioServico.js"></script>

    <script type="module" src="/static/JS/RF008A-mais.js"></script>

    <script src="/static/JS/scrollreveal.js"></script>

    <!-- Link para o usúario receber notificações -->

    <script defer type="module"
src="/static/JS/notificacoes.js"></script>



    <!-- Logo do Easy Request na guia-->

    <link rel="shortcut icon" type="imagex/png"
href="/static/IMG/TASK request.svg">

    <title>Detalhes do Encaminhamento</title>

</head>

<body>
```

```
    <nav>
        <section class="container-header">
            <div class="container-header_btn">
                <button class="container-header_btn-voltar"
onclick="funcVoltar()">
                    <i class="fa-solid fa-arrow-left"></i> Voltar
                </button>
            </div>
        </section>
    </nav>

    <main class="conteudo-principal">
        <section class="detalhes-solicitacao">
            <div class="detalhes-solicitacao__cabecalho">
                <h1 class="detalhes-solicitacao__titulo">{{
campo_detalhes_encaminhamento[0] }} - {{
campo_detalhes_encaminhamento[1] }}</h1>
                <p class="detalhes-solicitacao__autor">{{
campo_detalhes_encaminhamento[2] }} | {{
campo_detalhes_encaminhamento[3] }}</p>
            </div>
            <hr class="detalhes-solicitacao__divisor">
            <div class="detalhes-solicitacao__corpo">
                <p class="detalhes-solicitacao__rotulo-
descricao">Descrição da Solicitação</p>
```

```
                <p class="detalhes-solicitacao__texto-descricao">{{
campo_detalhes_encaminhamento[4] }}</p>

                {% if campo_detalhes_encaminhamento[7] %}

                <img src="{{campo_detalhes_encaminhamento[7]}}"
alt="Imagem que ilustra a solicitação">

                {%else%}

                {% endif %}

                <p class="detalhes-solicitacao__rotulo-
descricao">Observação do funcionario</p>

                <p class="detalhes-solicitacao__texto-descricao">{{
campo_detalhes_encaminhamento[5] }}</p>

                {% if campo_detalhes_encaminhamento[8] %}

                <img src="{{campo_detalhes_encaminhamento[8]}}"
alt="Imagem que ilustra a solicitação">

                {%else%}

                {% endif %}

                <section class="container-btn">

                    <button class="botao botao-cadastro"
onclick="finalizar_encaminhamento_supervisor({{campo_id_encaminhamen
to}})">Finalizar</button>

                    <button class="botao botao-reencaminhar"
onclick="reencaminhar_servico({{campo_detalhes_encaminhamento[6]}})"
>Reencaminhar</button>

                </section>

            </div>

        </section>
```

```
    </main>


    <!-- Modal -->
    <div id="modal" class="modal">
        <div class="modal__conteudo" id="modal-conteudo">
        </div>
    </div>


</body>
</html>
```

O CSS estiliza as cores e organiza os cards dos serviços no dispositivo. Além de determinar tamanho de fontes, imagem e dos cards. O código a seguir também abrange as medias, que fazem com que a página seja responsiva em diversos aplicativos.

```
/* Estilo para o elemento body */
body {
    overflow-x: hidden;
    height: auto;
}


/* Estilo para links */
```

```css
a {
    text-decoration: none;
}

/* Estilo para o elemento nav */
nav {
    width: 100vw;
    height: 9vh;
    display: flex;
    justify-content: space-between;
    align-items: flex-end;
    margin-top: 1.5rem;
}

/* Estilo para botões no cabeçalho */
.container-header_btn {
    display: flex;
    justify-content: center;
    align-items: center;
}
```

```css
/* Estilo específico para o botão de voltar */
.container-header_btn-voltar {
    margin: 1rem;
    padding: 0.5rem;
    width: 10rem;
    border-color: var(--color02);
    border-radius: 5rem;
    background: none;
    font-weight: 700;
    transition-duration: 0.2s;
}

/* Estilo para ícone dentro do botão de voltar */
.container-header_btn-voltar i {
    margin-right: 0.5rem;
}

/* Mudança de cor do botão ao passar o mouse */
.container-header_btn-voltar:hover {
    border-color: none;
    background-color: var(--color02);
    color: var(--color01);
}
```

```css
/* Estilo para a seção de detalhes da solicitação */
.detalhes-solicitacao {
    width: 100vw;
    min-height: 75vh;
    max-height: 75vh;
    overflow-y: auto; /* Adiciona rolagem vertical */
    overflow-x: hidden; /* Esconde a rolagem horizontal */
    height: auto;
    background-color: var(--color06);
    margin-bottom: 0;
    position: absolute;
    bottom: 0;
    left: 0;
    border-top-left-radius: 2rem;
    border-top-right-radius: 2rem;
    padding: 1.5rem;
    display: flex;
    flex-direction: column;
    align-content: center;
    justify-content: space-evenly;
    box-shadow: 1px -3px 10px rgba(0, 0, 0, 0.5);
    overflow-wrap: break-word;
```

```
    overflow-y: auto; /* Habilita a rolagem vertical */
}
/* estilo da barra de rolagem */
::-webkit-scrollbar {
    background-color: transparent;
}

/* Estilo para o cabeçalho da seção de detalhes */
.detalhes-solicitacao__cabecalho {
    text-align: center;
}

/* Estilo para o título na seção de detalhes */
.detalhes-solicitacao__titulo {
    font-size: 3rem;
    color: var(--color05);
    margin-bottom: 0.5rem;
    overflow-wrap: break-word;
}

/* Estilo para o autor da solicitação */
.detalhes-solicitacao__autor {
    font-size: 3.5rem;
```

```
    color: var(--color01);
    overflow-wrap: break-word;
}

/* Estilo para o divisor na seção de detalhes */
.detalhes-solicitacao__divisor {
    border: none;
    border-top: 2px solid var(--color05);
    margin: 1.8rem 0;
}

/* Estilo para o corpo da seção de detalhes */
.detalhes-solicitacao__corpo {
    padding: 0 3rem;
    font-size: 2.3rem;
    display: flex;
    flex-direction: column;
    align-items: center;
    text-align: justify;
    overflow-wrap: break-word;
}

/* Estilo para o rótulo da descrição */
```

```
.detalhes-solicitacao__rotulo-descricao {
    font-weight: bold;
    color: var(--color01);
    font-size: 2rem;
    overflow-wrap: break-word;
}

/* Estilo para o texto da descrição */
.detalhes-solicitacao__texto-descricao {
    width: 100%;
    padding: 0 2rem;
    margin-bottom: 1rem;
    line-height: 1.6;
    color: var(--color04);
    text-overflow: ellipsis;
    font-size: 1.7rem;
    overflow-wrap: break-word;
          text-align: center;
}

/* Estilo para imagens dentro do corpo da seção de detalhes */
.detalhes-solicitacao__corpo img {
    align-items: center;
```

```
}

/* Estilo para imagens na seção de detalhes */
.detalhes-solicitacao img {
    max-width: 30rem;
    max-height: 60rem;
    display: block;
    height: auto;
    object-fit: cover;
    border-radius: 0.5rem;
    margin-bottom: 1rem;
}

/* Estilos para o contêiner de botões */
.container-btn {
    width: 100vw;
    height: auto;
    display: flex;
    align-items: center;
    gap: 1rem;
    justify-content: center;
}
```

```css
/* Estilo base para botões */

.botao {

    display: inline-block;

    width: 25rem;

    padding: 1.2rem;

    text-align: center;

    border-radius: 2rem;

    font-size: 1.6rem;

    font-weight: bold;

    text-decoration: none;

    transition: all 0.3s ease-in-out;

    cursor: pointer;

    margin: 0 auto;

    display: flex;

    align-items: center;

    justify-content: center;

}


/* Estilo específico para o botão de reencaminhar */

.botao-reencaminhar {

    background-color: var(--color05);

    border: 2px solid var(--color05);

    color: var(--color01);
```

```
}

/* Estilo para o botão de reencaminhar ao passar o mouse */
.botao-reencaminhar:hover {
    background-color: transparent;
    border: 2px solid var(--color05);
    color: var(--color05);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.00);
}

/* Estilo específico para o botão de excluir */
.botao-excluir {
    background-color: var(--colorRed);
    color: var(--color01);
    border: none;
}

/* Estilo para o botão de excluir ao passar o mouse */
.botao-excluir:hover {
    background-color: #a90f0f;
    border: none;
}
```

```
/* Estilo para o botão de cadastro */
.botao-cadastro {
    background-color: transparent;
    border: 2px solid var(--color03);
    color: var(--color03);
}

/* Estilo do botão de cadastro ao passar o mouse */
.botao-cadastro:hover {
    background-color: var(--color03);
    color: var(--color06);
    box-shadow: 0px 8px 15px rgba(224, 223, 163, 0.2);
    transform: scale(1.00);
}

/* Estilo para o contêiner de botões */
.container-botao {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: row;
    gap: 2rem;
```

```
}


/* Estilos para o modal */
.modal {
    display: none;
    position: fixed;
    z-index: 1;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: auto;
    background-color: rgb(0, 0, 0);
    background-color: rgba(0, 0, 0, 0.4);
}


/* Estilo para o conteúdo do modal */
.modal__conteudo {
    background-color: #fefefe;
    margin: 15% auto;
    padding: 2rem;
    border: 1px solid #888;
    width: 80%;
```

```
}

.imagens_modal{
    list-style: none;
}

.imagens_modal img{
    width: 40%;
}

/* Estilo para o botão de fechar no modal */
.modal__fechar {
    color: #aaa;
    float: right;
    font-size: 2.8rem;
    font-weight: bold;
}

/* Estilo para o botão de fechar ao passar o mouse */
.modal__fechar:hover,
.modal__fechar:focus {
    color: black;
    text-decoration: none;
```

```
    cursor: pointer;
}


/* Estilos adicionais para o conteúdo do modal */
.modal__conteudo {
    background-color: #fff;
    border-radius: 0.8rem;
    box-shadow: 0 0.4rem 2rem rgba(0, 0, 0, 0.1);
    padding: 2rem;
}


/* Estilo para títulos h2 dentro do modal */
.modal__conteudo h2 {
    font-family: 'Georgia', serif;
    font-size: 2.4rem;
    color: #333;
    margin-top: 1.5rem;
    margin-bottom: 1rem;
    border-bottom: 0.2rem solid #eaeaea;
    padding-bottom: 0.5rem;
}


/* Estilo para subtítulos h3 dentro do modal */
```

```
.modal__conteudo h3 {
    font-family: 'Georgia', serif;
    font-size: 2rem;
    color: #444;
    margin-top: 1.5rem;
    margin-bottom: 0.5rem;
}

/* Estilo para listas não ordenadas dentro do modal */
.modal__conteudo ul {
    list-style-type: disc;
    margin-left: 2rem;
    color: #555;
}

/* Estilo para parágrafos dentro do modal */
.modal__conteudo p {
    font-size: 1.4rem;
    font-weight: bold;
}

/* Estilo para itens de listas dentro do modal */
.modal__conteudo ul li {
```

```
    margin-bottom: 0.5rem;
    font-size: 1.4rem;
}

/* Estilo para separadores (hr) dentro do modal */
.modal__conteudo hr {
    border: 1px solid #eaeaea;
    margin: 20px 0;
}

/* --------------------- MEDIA ------------------------ */
/* Responsividade para telas menores */
@media (max-width: 700px) {
    /* Estilo para imagens no corpo da seção de detalhes */
    .detalhes-solicitacao__corpo img {
        align-items: center;
    }

    /* Estilo para imagens na seção de detalhes */
    .detalhes-solicitacao img {
        max-width: 30rem;
        height: auto;
        border-radius: 0.5rem;
```

```
        margin-bottom: 1rem;
    }

    /* Estilo para o título na seção de detalhes */
    .detalhes-solicitacao__titulo {
        font-size: 1.5rem;
    }

    /* Estilo para a seção de detalhes */
    .detalhes-solicitacao {
        max-width: 70rem;
    }

    /* Estilo para o contêiner de botões */
    .container-botao {
        display: flex;
        justify-content: center;
        align-items: center;
        flex-direction: column;
        gap: 2rem;
    }

    /* Estilo para títulos na seção de detalhes */
```

```
    .detalhes-solicitacao__titulo,
    .detalhes-solicitacao__rotulo-descricao {
        font-size: 2rem;
    }

    /* Estilo para o autor e texto da descrição */
    .detalhes-solicitacao__autor,
    .detalhes-solicitacao__texto-descricao {
        font-size: 1.8rem;
    }
    .detalhes-solicitacao__texto-descricao {
        max-width: 40rem;
    }

    /* Estilo para o contêiner de botões */
    .container-btn {
        width: 100vw;
        height: auto;
        display: flex;
        align-items: center;
        gap: 1rem;
        justify-content: center;
        flex-direction: column;
```

```
    }
}


/* Estilo para ocultar elementos no carregamento */
#container-header,
#detalhes-solicitacao {
    visibility: hidden;
}
```

O JS nesta página executa a função ‘finalizar_encaminhamento_supervisor’ onde o administrador pode encerrar o serviço salvando os dados no banco do sistema. Ele contém também as transições da página e mensagem de sucesso e erro de acordo com as ações do usuário

```
// Inicializa o ScrollReveal
ScrollReveal().reveal('.container-header', {
    duration: 600,
    origin: 'top', // O cabeçalho surge de cima
    distance: '50px',
    delay: 200,
    afterReveal: function (el) {
        el.style.opacity = 1; // Define a opacidade após a revelação
    }
});
```

```
ScrollReveal().reveal('.detalhes-solicitacao', {
    duration: 600,
    origin: 'bottom', // O conteúdo principal surge de baixo
    distance: '50px',
    delay: 200,
    afterReveal: function (el) {
        el.style.opacity = 1; // Define a opacidade após a revelação
    }
});

// Função para ir para a tela de finalização
function funcVoltar() {
    window.location.href = '/RF008';
}

// Ativando a função funcVoltar
window.funcVoltar = funcVoltar;

// Função para ir para a tela de encaminhar solicitação
function reencaminhar_servico(id_solicitacao) {
    window.location.href = `/RF005/${id_solicitacao}`;
}
```

```
// Ativando a função reencaminhar_servico
window.reencaminhar_servico = reencaminhar_servico;


// Função que permite que o supervisor finalize o serviço
oficialmente no sistema
function finalizar_encaminhamento_supervisor(id_encaminhamento) {
    $.ajax({
        url: `/finalizacao-encaminhamento-
supervisor/${id_encaminhamento}`,
        type: 'GET',
        success: function(){
            Swal.fire({
                title: "Serviço Finalizado",
                text: "Escolha se deseja ver o relatório do
serviço!",
                icon: "success",
                showCancelButton: true,
                confirmButtonColor: "#04A61C",
                cancelButtonColor: "#6d8ce8",
                confirmButtonText: "Terminar",
                cancelButtonText: "Ver Relatório",
                customClass: {
                    title: 'custom-title', // Classe para o título
```

```
                    content: 'custom-content', // Classe para o
conteúdo
                    confirmButton: 'custom-button', // Classe para o
botão de confirmação
                    cancelButton: 'custom-button' // Classe para o
botão de cancelamento
                }
            }).then((result) => {
                if (result.isConfirmed) {
                    window.location.href = '/RF008';
                } else if(result.dismiss ===
Swal.DismissReason.cancel) {
                    criarRelatorio(id_encaminhamento)
                }
            });
        },
        error: function(mensagem){
            Swal.fire({
                icon: "error",
                title: "Oops...",
                text: mensagem['responseText'],
                showConfirmButton: false,
                timer: 3500
              });
```

```
        }
    });
}


// Ativando a função finalizar_encaminhamento_supervisor
window.finalizar_encaminhamento_supervisor =
finalizar_encaminhamento_supervisor;



// CSS para as classes personalizadas
const style = document.createElement('style');
style.innerHTML = `
    .custom-title {
        font-size: 24px; /* Tamanho do título */
    }
    .custom-content {
        font-size: 20px; /* Tamanho do texto */
    }
    .custom-button {
        font-size: 16px; /* Tamanho dos botões */
    }
`;
document.head.appendChild(style);
```

O python cria a rota para a página e cria a função de mostrar na tela os serviços encaminhados para ser finalizado.

```
# Criando a rota para a tela de detalhes da
finalização(Administrador)

@app.route("/RF008A/<id_encaminhamento>")

def pg_detalhes_finalizacao(id_encaminhamento): # Função que
renderiza a tela de detalhes da finalização(Administrador)

    if "usuario" in session:

        encaminhamento = Encaminhamento()

        detalhes_ecaminhamento =
encaminhamento.mostrar_detalhes_encaminhamento(id_encaminhamento)

        return render_template("RF008A-mais.html",
campo_detalhes_encaminhamento = detalhes_ecaminhamento,
campo_id_encaminhamento = id_encaminhamento)

    else:

        return redirect("/")



# Criando a rota que finaliza o serviço(Administrador)

@app.route("/finalizacao-encaminhamento-
supervisor/<id_encaminhamento>")

def finalizacao_encaminhamento_supervisor(id_encaminhamento): #
Criando a rota que finaliza o serviço oficialmente no banco de
dados(Administrador)

    encaminhamento = Encaminhamento()
```

```
    retorno = encaminhamento.finalizacao_encaminhamento_supervisor(id_encaminhamento)

    print(retorno)

    if retorno == 'Você já finalizou esse serviço':

        return 'Você já finalizou esse serviço', 500

    else:

        return jsonify({'mensagem':'Serviço Finalizado'}), 200
```

```
 # Função que mostra os detalhes do serviço que foi encaminhado para o funcionário da manutenção

    def mostrar_detalhes_encaminhamento(self, id_encaminhamento):

        myBD = Connection.conectar()


        mycursor = myBD.cursor()


        self.id_encaminhamento = id_encaminhamento


        print(id_encaminhamento)


        mycursor.execute(f"SELECT s.id_sala, sv.nome, f.nome, status_final, sol.descricao, adendo, enc.id_solicitacao, sol.foto,
```

```
enc.foto_finalizacao FROM tb_salas s, tb_servicos sv,
tb_funcionarios f, tb_encaminhamentos enc, tb_solicitacoes sol WHERE
sol.id_solicitacao = enc.id_solicitacao AND sv.id_servico =
sol.id_servico AND s.id_sala = sol.id_sala AND f.CPF_funcionario =
sol.CPF_funcionario AND enc.id_encaminhamento =
{id_encaminhamento};")


        mostra_encaminhamentos = mycursor.fetchone()


        return mostra_encaminhamentos
```

```
  # Função que permite que o Administrador finalize o serviço
oficialmente no sistema

    def finalizacao_encaminhamento_supervisor(self,
id_encaminhamento):

        myBD = Connection.conectar()


        mycursor = myBD.cursor()

        data_relatorio = datetime.now().strftime('%Y-%m-%d
%H:%M:%S')



        mycursor.execute(f"""SELECT status_finalizacao FROM
tb_encaminhamentos WHERE id_encaminhamento =
{id_encaminhamento};""")
```

```
        retorno = mycursor.fetchone()[0]


        if retorno == '1':
            return "Você já finalizou esse serviço"
        else:
            mycursor.execute(f"UPDATE tb_encaminhamentos SET
status_finalizacao = TRUE WHERE id_encaminhamento =
{id_encaminhamento};")


            myBD.commit()


            mycursor.execute(f"INSERT INTO tb_relatorios
(id_encaminhamento, data_relatorio) VALUES ({id_encaminhamento},
'{data_relatorio}')")


            myBD.commit()


            return True
```

Abaixo os códigos do relatório em python sobre a rota e funções.

```
# Criando a rota que gera o relatório de serviço
@app.route("/RF009/<id_encaminhamento>")
def criar_relatorio(id_encaminhamento): # Função que gera um
relatório de serviço baseado em todas baseado nas informações da
solicitação e do encaminhamento
```

```
    if "usuario" in session:

        encaminhamento = Encaminhamento()

        info_relatorio =
encaminhamento.dados_relatorio(id_encaminhamento)

        return jsonify(info_relatorio), 200

    else:

        return redirect("/")
```

```
 # Função que pega os dados que precisam ser colocados no relatório

    def dados_relatorio(self, id_encaminhamento):



        self.id_encaminhamento = id_encaminhamento



        myBD = Connection.conectar()



        mycursor = myBD.cursor()



        mycursor.execute(f"""SELECT DATE_FORMAT(r.data_relatorio,
'%d/%m/%Y %H:%i:%s'), sol.id_solicitacao, enc.id_encaminhamento,
f1.nome,

                         sol.CPF_funcionario, IF(fn.id_funcao IS
NULL, 'Administrador', fn.nome) AS nome_funcao, f1.email, sv.nome,
sol.id_sala,
```

```
                            s.bloco, sol.descricao,
DATE_FORMAT(sol.data_inicio, '%d/%m/%Y %H:%i:%s'), sol.foto,
f2.nome, enc.CPF_funcionario, enc.urgencia,

                            DATE_FORMAT(enc.data_inicio_servico,
'%d/%m/%Y %H:%i:%s'), DATE_FORMAT(enc.data_termino_servico,
'%d/%m/%Y %H:%i:%s'), enc.adendo, enc.foto_finalizacao

                            FROM tb_relatorios r

                            JOIN tb_encaminhamentos enc ON
r.id_encaminhamento = enc.id_encaminhamento

                            JOIN tb_solicitacoes sol ON
sol.id_solicitacao = enc.id_solicitacao

                            JOIN tb_funcionarios f1 ON
f1.CPF_funcionario = sol.CPF_funcionario

                            LEFT JOIN tb_funcoes fn ON fn.id_funcao =
f1.id_funcao

                            JOIN tb_funcionarios f2 ON
f2.CPF_funcionario = enc.CPF_funcionario

                            JOIN tb_servicos sv ON sv.id_servico =
sol.id_servico

                            JOIN tb_salas s ON s.id_sala = sol.id_sala

                            WHERE enc.id_encaminhamento =
{id_encaminhamento};""")


        dados_relatorio = mycursor.fetchone()


        print(dados_relatorio)
```

```
        return dados_relatorio
```

Apêndice O – Histórico de Relatórios

Neste apêndice é responsável pela tela de histórico de relatórios, que quando o supervisor de manutenção finaliza o serviço é criado um relatório e logo depois é salvo numa página que é o histórico de relatório em que lá ele pode acessar o relatório novamente e se preferir pode baixar o relatório.

O HTML que tem a função de marcar os códigos e distribuir os elementos da página com todas as separações, o header insere as informaçõe que estão no cabeçalho do sistema como botão de voltar com a função de JavaScript(funcVoltar()) e também a barra de pesquisa que permite o supervisor pesquisar qual relatório ele vai querer ver, e no main existe o container que vai guardar as informações dos relatórios.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <!-- Definindo a codificação de caracteres e a responsividade --
>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <!-- Links para os estilos CSS globais e específicos da página -
->
    <link rel="stylesheet" href="/static/CSS/global.css">
    <link rel="stylesheet" href="/static/CSS/RF010-
historicoRelatorio.css">
    <!-- Importação do JS do site -->
```

```
    <script src="/static/JS/RF010-historicoRelatorio.js"></script>

    <!-- link JS biblioteca scrollrevealjs -->

    <script src="https://unpkg.com/scrollreveal"></script>

    <!-- Link para a biblioteca de ícones Font Awesome -->

    <script src="https://kit.fontawesome.com/e9b74fb3c4.js"
crossorigin="anonymous"></script>

    <!-- Biblioteca jQuery para facilitar o uso de Ajax e
manipulação de DOM -->

    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.j
s"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/dompurify/2.3.10/purify.
min.js"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/html2c
anvas.min.js"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.mi
n.js"></script>

    <!-- Links para o JavaScript -->

    <script defer src="/static/JS/RF009-
relatorioServico.js"></script>

    <script type="module" src="/static/JS/RF010-
historicoRelatorio.js"></script>

    <!-- Logo do Easy Request na guia-->

    <link rel="shortcut icon" type="imagex/png"
href="/static/IMG/TASK request.svg">

    <script
src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
```

```
    <!-- Link para o usúario receber notificações -->
    <script defer type="module"
src="/static/JS/notificacoes.js"></script>


    <!-- Título da página -->
    <title>Histórico de Relatórios</title>
</head>
<body>
    <!-- Cabeçalho da página -->
    <header class="cabecalho" id="cabecalho">
        <button class="cabecalho__botao-voltar"
onclick="funcVoltar()"><i class="fa-solid fa-arrow-
left"></i>Voltar</button>
        <!-- Título do cabeçalho -->
        <h1 class="cabecalho__titulo"><span>Histórico</span> de
serviços</h1>


        <!-- Barra de pesquisa no cabeçalho -->
        <nav class="cabecalho__pesquisa">
            <input type="text" class="cabecalho__input"
placeholder="Pesquisar..." id="pesquisa">
            <button class="cabecalho__botao" id="botaoPesquisa">
                <i class="cabecalho__icone fa-solid fa-magnifying-
glass"></i>
            </button>
        </nav>
    </header>
```

```
    <!-- Conteúdo principal da página -->
    <main class="conteudo" id="conteudo-historico">
    </main>
    <!-- Modal -->
    <div id="modal" class="modal">
        <div class="modal__conteudo" id="modal-conteudo">
            <span class="modal__fechar"
id="fecharModal">&times;</span>
        </div>
    </div>
</body>
</html>
```

No CSS desta página, está sendo alterado o botão que tem a função de voltar para a página anterior(.cabecalho__botao-voltar), o body que será onde vai inserir as informações dos relatórios também está sendo alterado para deixar o sistema intuitivo e fácil usabilidade, e com o @media está deixando o sistema adaptado para várias telas.

```
/* Estilo do btn cabecalho */
.cabecalho__botao-voltar{
    display: flex;
    text-align: center;
    margin: 0.5rem 0rem 1rem 0rem;
    color: var(--color01);
    padding: 0.5rem;
```

```
    width: 10rem;
    border: 1px solid var(--color01);
    border-radius: 5rem;
    background: none;
    font-weight: 700;
    transition: background-color 0.2s, color 0.2s, border-color
0.2s;
    justify-content: center;
}

.cabecalho__botao-voltar i {
    margin-right: 0.5rem;


}

.cabecalho__botao-voltar:hover {
    border-color: var(--color05);
    background-color: var(--color05);
}

/* Estilo do cabeçalho da página, centralizando texto e adicionando
estilo visual */
html, body {
    overflow-x: hidden;
    height: 100%; /* Garante que o body ocupe toda a altura da
página */
}
```

```
body {
    position: relative; /* Para conter o conteúdo e trabalhar com o hack */
    overflow-x: hidden; /* Remove a barra de rolagem horizontal */
}

body::before {
    content: '';
    position: absolute;
    top: 0;
    bottom: 0;
    right: -1.5rem; /* Ajusta essa margem para controlar a largura invisível da rolagem */
    width: 1.5rem; /* Largura da barra de rolagem invisível */
    background: transparent; /* Mantém a área invisível */
    overflow-y: scroll; /* Mantém a funcionalidade de rolagem */
    overflow-x: hidden;
}

/* Cabecalho posicionado na parte de cima */
.cabecalho {
    text-align: center;
    padding: 2rem 2rem 3rem 2rem;
    background-color: var(--color06);
    color: var(--color01);
```

```
    border-bottom-left-radius: 2rem;
    border-bottom-right-radius: 2rem;
    position: sticky;
    top: 0;
    box-shadow: 0px 4px 10px #00000061;
}

/* Estilo do título dentro do cabeçalho, ajustando tamanho e margem
*/
.cabecalho__titulo {
    font-size: 2.4rem;
    margin: 0;
    font-weight: 700;
}

h1 span{
    color: var(--color05);
}

/* Estilo da barra de pesquisa no cabeçalho, centralizando elementos
*/
.cabecalho__pesquisa {
    margin-top: 1rem;
    display: flex;
    justify-content: center;
}
```

```
/* Estilo do campo de entrada da pesquisa, ajustando preenchimento e bordas */
.cabecalho__input {
    padding: 1rem;
    border: none;
    border-radius: 0.4rem 0 0 0.4rem;
    width: 70%;
}

/* Estilo do botão de pesquisa, definindo cor de fundo e bordas arredondadas */
.cabecalho__botao {
    background-color: var(--color01);
    border: none;
    padding: 1rem;
    border-radius: 0 0.4rem 0.4rem 0;
    cursor: pointer;
}

.container-botao{
    width: 100%;
    text-align: center;
}

/* Estilo do ícone dentro do botão, definindo largura */
```

```
.cabecalho__icone {
    width: 2rem;
}



/* Estilo do conteúdo principal, configurando layout em grid */
.conteudo {
    display: grid;
    gap: 2rem;
    padding: 2rem;
    grid-template-columns: 1fr;

}

.conteudo h3{
    text-align: center;
    color: var(--color06);
    grid-column: span 3;
}

/* Estilo do cartão que contém informações de serviços, com fundo e bordas */
.conteudo__card {
    display: flex;
    background-color: var(--color04);
    border-radius: 1rem;
```

```
    padding: 1rem;
    color: var(--color06);
    transition-duration: 0.2s;
}

.conteudo__card:hover{
    background-color: #c9c9c9;
}

/* Estilo da imagem dentro do cartão, tornando-a circular */

.conteudo__imagem {
    border-radius: 50%;
    min-width: 6rem;
    max-height: 6rem;
    margin-right: 1rem;
    object-fit: cover;
}

/* Estilo da seção de informações do cartão, organizando conteúdo em
coluna */
.conteudo__informacao {
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: flex-start
```

```
}

/* Estilo do nome dentro do cartão, ajustando tamanho da fonte */
.conteudo__nome {
    margin: 0;
    font-size: 2.5rem;
    border-bottom: solid var(--color05);
}

/* Estilo da descrição dentro do cartão, definindo tamanho da fonte e margem */
.conteudo__descricao {
    font-size: 1.7rem;
    margin: 1rem 0;
    word-break: break-word;

}

/* Responsividade: configura layout para telas maiores que 600px */
@media (min-width: 600px) {
    .conteudo {
        grid-template-columns: 1fr 1fr;
    }

}
```

```
/* Responsividade: configura layout para telas maiores que 900px */
@media (min-width: 900px) {
    .conteudo {
        grid-template-columns: 1fr 1fr 1fr;
    }

    .cabecalho__titulo {
        font-size: 3.4rem;
    }
}



/* Estilo das datas no cartão, ajustando tamanho da fonte */
.conteudo__datas {
    font-size: 1.6rem;
}

/* Estilo do botão "Ver mais", definindo largura e aparência */
.conteudo__botao {
    width: 100%;
    background-color: var(--color03);
    margin-top: 1rem;
    border: none;
    padding: 0.5rem 2rem;
    color: var(--color06);
    cursor: pointer;
```

```
    border-radius: 0.4rem;
    align-self: flex-start;
    font-weight: 700;
    transition-duration: 0.2s;
}

.conteudo__botao:hover{
    background-color: var(--color05);
}




.modal {
    display: none; /* Escondido por padrão */
    position: fixed;
    z-index: 1;
    left: 0;
    top: 0;
    width: 100%; /* Largura total */
    height: 100%; /* Altura total */
    overflow: auto; /* Ativar rolagem se necessário */
    background-color: rgb(0,0,0); /* Cor de fundo */
    background-color: rgba(0,0,0,0.4); /* Fundo preto com opacidade
*/
}
```

```css
.modal__conteudo {
    background-color: #fefefe;
    margin: 15% auto; /* Margens automáticas para centralizar */
    padding: 2rem;
    border: 1px solid #888;
    width: 80%; /* Largura do modal */
}

.imagens_modal{
    list-style: none;
}

.imagens_modal img{
    width: 40%;
}

.modal__fechar {
    color: #aaa;
    float: right;
    font-size: 2.8rem;
    font-weight: bold;
}

.modal__fechar:hover,
.modal__fechar:focus {
    color: black;
```

```
    text-decoration: none;
    cursor: pointer;
}



/* Estilos para o modal */
.modal__conteudo {
    background-color: #fff; /* Fundo branco */
    border-radius: 0.8rem; /* Bordas arredondadas */
    box-shadow: 0 0.4rem 2rem rgba(0, 0, 0, 0.1); /* Sombra suave */
    padding: 2rem; /* Espaçamento interno */
}
/* Estilo para os títulos h2 */
.modal__conteudo h2 {
    font-family: 'Georgia', serif; /* Fonte clássica */
    font-size: 2.4rem; /* Tamanho do texto */
    color: #333; /* Cor escura */
    margin-top: 1.5rem; /* Margem superior */
    margin-bottom: 1rem; /* Margem inferior */
    border-bottom: 0.2rem solid #eaeaea; /* Linha abaixo */
    padding-bottom: 0.5rem; /* Espaçamento interno abaixo */
}

/* Estilo para os subtítulos h3 */
.modal__conteudo h3 {
    font-family: 'Georgia', serif; /* Fonte clássica */
```

```
    font-size: 2rem; /* Tamanho do texto */
    color: #444; /* Cor um pouco mais clara */
    margin-top: 1.5rem; /* Margem superior */
    margin-bottom: 0.5rem; /* Margem inferior */
}

/* Estilo para listas não ordenadas */
.modal__conteudo ul {
    list-style-type: disc; /* Marcadores em forma de disco */
    margin-left: 2rem; /* Margem à esquerda */
    color: #555; /* Cor do texto */
}
.modal__conteudo p{
    font-size: 1.4rem;
    font-weight: bold;
}
/* Estilo para itens da lista */
.modal__conteudo ul li {
    margin-bottom: 0.5rem; /* Espaçamento entre itens */
    font-size: 1.4rem;
}

/* Estilo para separadores (hr) */
.modal__conteudo hr {
    border: 1px solid #eaeaea; /* Linha sutil */
    margin: 20px 0; /* Margem vertical */
```

```
}
.container-header_btn{
    display: flex;
    justify-content: center;
    align-items: center;
}

.container-header_btn-voltar{
    margin: 1rem;
    padding: 0.5rem;
    width: 10rem;
    border-color: var(--color02);
    border-radius: 5rem;
    background: none;
    font-weight: 700;
    cursor: pointer;
    transition-duration: 0.2s;
}

.container-header_btn-voltar i{
    margin-right: 0.5rem;
 }

 /* Mudança de Cor do botão ao passar o cursor por cima */
.container-header_btn-voltar:hover{
    border-color: none;
```

```
    background-color: var(--color02);
    color: var(--color01);
}
```

No JavaScript, as funções desta página está as principais funções que é a que faz o botão voltar para a página anterior (funcVoltar()) os efeitos da página (ScrollReveal), a função que obtém o modal (const modal) e a função em AJAX que pega os dados que o Python está trazendo e inserindo no HTML através do jinja (function(historico)).

```
document.addEventListener('DOMContentLoaded', function() {
    // Inicializa o ScrollReveal
    window.sr = ScrollReveal({
        reset: true // Efeito se repete sempre que o elemento
reaparece na tela
    });

    // Aplica o efeito de revelação ao cabeçalho
    sr.reveal('.cabecalho', {
        duration: 600,
        origin: 'top',
        distance: '50px',
        delay: 200
    });

});
```

```
// Obtém o modal
const modal = document.getElementById("modal");

// Obtém o botão "Ver mais"
const botaoVerMais = document.querySelector(".conteudo__botao");

// Função para ir para a tela inicial do Administrador
function funcVoltar() {
    window.location.href = '/tl-administrador';
}

// Obtém o Container do histórico
var campo_historico = document.querySelector('#conteudo-historico');

// Função que mostra todos os itens do histórico
function retornaHistorico() {
    if (document.querySelector('#pesquisa').value == ''){
        var pesquisa = null;
    }else{
        var pesquisa = document.querySelector('#pesquisa').value;
    }

    $.ajax({
      url: `/retorna-historico/${pesquisa}`,
      type: 'GET',
```

```
      success: function(historico){
        console.log(historico)
        campo_historico.innerHTML = '';
        if (historico.length == 0) {
            campo_historico.innerHTML = '<h3>Não há serviços
finalizados.</h3>';
        }else{
            for (let x = 0; x < historico.length; x++) {
                if (historico[x][0]) {
                    var imagem_funcionario = `<img
src="${historico[x][0]}" alt="Foto de Perfil"
class="conteudo__imagem">`
                }else{
                    var imagem_funcionario = '<img
src="https://st4.depositphotos.com/11574170/25191/v/450/depositphoto
s_251916955-stock-illustration-user-glyph-color-icon.jpg" alt="Foto
de Perfil" class="conteudo__imagem">'
                }
                if(historico[x][3]){
                   var descricao = `<p
class="conteudo__descricao"><b>Descrição:</b>
${historico[x][3]}</p>`
                }else{
                    var descricao =`<p
class="conteudo__descricao"><b>Descrição:</b> Não tem
descrição.</p>`
                }
                var conteudoHistorico = `<!-- Cartão de histórico de
serviço -->
```

```
                <div class="conteudo__card">
                    <!-- Imagem do profissional responsável pelo
serviço -->
                    ${imagem_funcionario}

                    <!-- Informações sobre o serviço realizado -->
                    <div class="conteudo__informacao">
                        <h2
class="conteudo__nome">${historico[x][1]}</h2>

                        <!-- Descrição do serviço -->
                        ${descricao}

                        <hr>

                        <!-- Datas do serviço -->
                        <p class="conteudo__datas"><b>Tarefa
atribuída em:</b> ${historico[x][4]}</p>
                        <p class="conteudo__datas"><b>Tarefa
finalizada em:</b> ${historico[x][5]}</p>

                        <!-- Botão para ver mais detalhes sobre o
serviço -->
                        <div class="container-botao">
                        <button class="conteudo__botao"
onclick="criarRelatorio(${historico[x][6]})">Ver mais</button>
                        </div>
                    </div>
```

```
                </div>
                `;
                campo_historico.innerHTML += conteudoHistorico;
            }
        }
      },
      error: function(){
          Swal.fire({
              icon: "error",
              title: "Oops...",
              text: "Erro no retorno do Histórico!",
              showConfirmButton: false,
              timer: 3500
            });
      }
  });
}

// Funções do botão de pesquisa
document.addEventListener('DOMContentLoaded', function () {
    var btnPesquisa = document.querySelector('#botaoPesquisa');

    // Verifica se o elemento #botaoPesquisa existe antes de
adicionar o event listener
    if (btnPesquisa) {
        btnPesquisa.addEventListener('click', function() {
```

```
            retornaHistorico();
        });
    }

    // Adiciona o evento de "keypress" ao campo de entrada, se
necessário
    var inputPesquisa = document.querySelector('#pesquisa');
    if (inputPesquisa) {
        inputPesquisa.addEventListener("keypress", function (event)
{
            if (event.key === "Enter") {
                retornaHistorico();
            }
        });
    }

    // Chama a função assim que a página carrega
    retornaHistorico();
});
```

o Python desta página está com a rota (RF/010) que existe a sua função no arquivo app.py(def pg_historico()) ela serve para renderizar a tela, e também no arquivo app.py existe uma outra função que também é ligado a essa tela que é a função (def_retorna_historico(pesquisa)) nesta função está as informações que vem do banco de dados através (encaminhamento.retorna_historico(pesquisa)) que esta no arquivo Encaminhamento.py que lá esta todas as informações que vem do banco

de dados através das conexões feitas e enviadas ao JavaScript que insere no HTML.

```
@app.route("/RF010")
def pg_historico(): # Função que renderiza a tela de relatórios de serviço
    if "usuario" in session:
        return render_template("RF010-historicoRelatorio.html")
    else:
        return redirect("/")


# Criando a rota que retorna os itens do histórico
@app.route("/retorna-historico/<pesquisa>")
def retorna_historico(pesquisa): # Função que retorna os itens do histórico de serviço, baseado no que o usuário pesquisou
    encaminhamento = Encaminhamento()
    historico = encaminhamento.retorna_historico(pesquisa)
    return jsonify(historico), 200
```

```
   def retorna_historico(self, pesquisa):
        self.pesquisa = pesquisa
        myBD = Connection.conectar()


        mycursor = myBD.cursor()


        if pesquisa == "null":
            pesquisa = ""
```

```
        mycursor.execute(f"SELECT f.foto, f.nome, fn.nome,
enc.adendo, DATE_FORMAT(enc.data_inicio_encaminhamento, '%d/%m/%Y'),
DATE_FORMAT(enc.data_termino_servico, '%d/%m/%Y'),
enc.id_encaminhamento FROM tb_funcionarios f, tb_funcoes fn,
tb_encaminhamentos enc WHERE f.id_funcao = fn.id_funcao AND
f.CPF_funcionario = enc.CPF_funcionario AND status_finalizacao =
TRUE AND f.nome LIKE '%{pesquisa}%' ORDER BY
DATE_FORMAT(enc.data_termino_servico, '%H:%i:%s') DESC;")


        historico = mycursor.fetchall()


        return historico
```

Apêndice P – Notificação

Este código JavaScript da notificação integra o sistema de notificações push utilizando o Firebase, permitindo que os usuários recebam alertas em tempo real. Inicialmente. Quando a página é carregada, o código solicita permissão ao usuário para receber notificações. Uma vez concedida a permissão, o sistema gera um token único para o dispositivo do usuário e o envia ao backend, permitindo que o backend envie notificações direcionadas ao usuário.

O código também fica monitorando a chegada de novas mensagens enquanto o site está em primeiro plano. Quando uma notificação chega, o título e a mensagem da notificação são extraídos e processados. Um som de notificação é reproduzido, e uma janela de notificação é exibida usando a biblioteca SweetAlert2 (Swal.fire). Além disso, a notificação possui um estilo personalizado, aparecendo no canto superior direito da tela. Ela é configurada para desaparecer automaticamente após 5 segundos.

```
// Importar as funções necessárias do Firebase
import { initializeApp } from
"https://www.gstatic.com/firebasejs/11.0.1/firebase-app.js";
import { getMessaging, getToken, onMessage } from
"https://www.gstatic.com/firebasejs/11.0.1/firebase-messaging.js";


// Adicionando as informações necessárias no objeto firebaseConfig
const firebaseConfig = {
    apiKey: "AIzaSyB_9zDaMKqg34_KJw443nDgcsKQwAZGc8I",
    authDomain: "easy-request-17b8a.firebaseapp.com",
    projectId: "easy-request-17b8a",
    storageBucket: "easy-request-17b8a.firebasestorage.app",
    messagingSenderId: "1044235475295",
    appId: "1:1044235475295:web:f995b81342895cc3433eea",
    measurementId: "G-3Z5196DP9F"
};


// Inicializar Firebase
const app = initializeApp(firebaseConfig);
const messaging = getMessaging(app);


// Variável global para armazenar o CPF do usuário
let usuarioCPF;
```

```
// Função para obter o CPF do usuário
fetch('/usuario_cpf')
    .then(response => response.json())
    .then(data => {
        usuarioCPF = data.usuario_cpf;
        console.log(usuarioCPF);  // Exibe o CPF do usuário no
console
        // Chamar a função para registrar o Service Worker
        setTimeout(solicitarPermissaoParaNotificacoes, 500);
    })
    .catch(error => console.error('Erro ao obter CPF do usuário:',
error));


// Solicitar permissão e obter o token do usuário
function solicitarPermissaoParaNotificacoes() {
    getToken(messaging, { vapidKey:
"BHPhiO7dBCSH9lMRIPNSNOvDCYRqGLE5bN388E9C9giGRdPZsymjGJ4qd76iAlHnslz
dlNNlRvpU4HeSkpOHI8E" })  // Substitua pela sua chave VAPID
        .then((currentToken) => {
            if (currentToken) {
                console.log("Token do usuário:", currentToken);
                // Enviar o token para o backend
                fetch('/salvar-token', {
                    method: 'POST',
                    headers: {
```

```
                        'Content-Type': 'application/json'
                    },
                    body: JSON.stringify({
                        token: currentToken,
                        usuario_cpf: usuarioCPF  // Envia o CPF do
usuário
                    })
                })
                .then(response => response.json())
                .then(data => {
                    console.log("Resposta do backend:", data);
                })
                .catch((error) => {
                    console.error("Erro ao enviar o token:", error);
                });

            } else {
                console.log("Nenhum token disponível. Solicitação de
permissão foi rejeitada.");
            }
        })
        .catch((err) => {
            console.log("Erro ao obter o token:", err);
```

```
            // Forçar a recarga da página para garantir que tudo
funcione corretamente
            solicitarPermissaoParaNotificacoes();
        });
}

// Ouvir as mensagens recebidas enquanto o site está em primeiro
plano
onMessage(messaging, (payload) => {
    console.log("Mensagem recebida no primeiro plano:", payload);

    const titulo = payload.notification.title || "Nova Notificação";
    const mensagem = payload.notification.body || "Você recebeu uma
nova mensagem.";

    const somNotificacao = new Audio('static/SONDS/livechat-
129007.mp3'); // Altere o caminho para o seu arquivo de som

    // Toca o som
    somNotificacao.play().catch(error => console.warn("Erro ao
reproduzir som:", error));

    // Tocar o som de notificação
    somNotificacao.play().then(() => {
        Swal.fire({
```

```
            title: titulo,
            text: mensagem,
            iconHtml: '<i class="fa fa-bell"></i>', // Ícone
personalizado
            customClass: {
                popup: 'swal2-notificacao' // Classe CSS para estilo
            },
            showConfirmButton: false, // Sem botão de confirmação
            timer: 5000, // Fecha automaticamente após 5 segundos
            position: 'top-end', // Notificação no canto superior
direito
            toast: true, // Estilo "toast"
            width: window.innerWidth <= 600 ? '60vw' : '30vw', //
Largura proporcional à tela, limitando a 60vw
            // Evento de clique na notificação (popup)
            willOpen: () => {
                const popup = Swal.getPopup();
                popup.style.cursor = 'pointer'; // Torna o cursor
"pointer"
                popup.addEventListener('click', () => {
                    // Redireciona para a URL fornecida
                    window.location.href = '/RF004';
                });
            }
        });
```

```
    }).catch((error) => {
        console.error("Erro ao tocar som de notificação:", error);
    });

});
```

Esse código registra um "Service Worker", que é um script que permite o recebimento de notificações mesmo quando o usuário não está com o site aberto. Ele tenta registrar o arquivo /firebase-messaging-sw.js e, se for bem-sucedido, exibe uma mensagem de sucesso no console. Se ocorrer um erro, ele exibe a mensagem de erro.

```
// Registrando o Service Worker do Firebase
navigator.serviceWorker.register('/firebase-messaging-sw.js')
    .then(function(registration) {
        console.log('Service Worker registrado com sucesso:',
registration);
    })
    .catch(function(error) {
        console.error('Erro ao registrar o Service Worker:', error);
    });
```

Apêndice Q – Códigos para Cadastro do Admistrador

Esse código é utilizado para cadastrar manualmente um administrador no sistema, inserindo seus dados (como CPF, nome, e-mail e senha) no banco de dados, com a senha sendo armazenada de forma segura por meio de criptografia.

```
# Importação das funções para conectar o MySQL e criar senhas criptografadas.
from conexao_SQL import Connection
from hashlib import sha256


# Função que cadastra o administrador no Banco de Dados de maneira semi-manual
def cadastrarAdm():
    try:
        myBD = Connection.conectar()


        mycursor = myBD.cursor()


        cpf = "463.129.468-00"
        nome = "Ana Beatriz Camassuti"
        email = "ana.franciscatto4419@gmail.com"
        senha = "ana123"
        sn = "SN123456"
        permissao = "administrador"
        senha_criptografada = sha256(senha.encode()).hexdigest()


        mycursor.execute(f"INSERT INTO tb_funcionarios (CPF_funcionario, nome, email, senha, SN, permissao) VALUES ('{cpf}','{nome}', '{email}', '{senha_criptografada}', '{sn}', '{permissao}');")


        myBD.commit()
```

```
        return True
    except:
        return False

if cadastrarAdm():
    print("Usuário cadastrado com Sucesso")
else:
    print("Erro ao cadastrar Administrador")
```

Apêndice R – Criar o Banco de Dados

O código tem como objetivo criar a estrutura do banco de dados do sistema, com tabelas para armazenar informações sobre funções dos funcionários, salas, serviços, solicitações, encaminhamentos e relatórios finais. Além disso, o código configura o acesso ao banco, criando um usuário e concedendo permissões para gerenciar os dados.

```
CREATE DATABASE bd_easyrequest;

#Selecionar o banco de Dados a ser utilizado
USE bd_easyrequest;

#Criar usuário para acessar o Banco de Dados
CREATE USER 'equipe_easyrequest'@'%' IDENTIFIED BY '4_b4t@t45_3_m31@';
GRANT ALL PRIVILEGES ON bd_easyrequest.* TO 'equipe_easyrequest'@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

```
#Criar tabela que armazena as funções
CREATE TABLE tb_funcoes (
 id_funcao INT AUTO_INCREMENT PRIMARY KEY,
 nome VARCHAR(100) NOT NULL,
 itens VARCHAR(100) NOT NULL
);

#Criar tabela que armazena as salas
CREATE TABLE tb_salas (
 id_sala VARCHAR(10) PRIMARY KEY,
 nome_sala VARCHAR(100) NOT NULL,
 bloco VARCHAR(2) NOT NULL
);

#Criar a tabela que armazena os serviços
CREATE TABLE tb_servicos (
 id_servico INT PRIMARY KEY AUTO_INCREMENT,
 nome VARCHAR(100) NOT NULL
);

#Criar a tabela que armazena os funcionários/ usuários do sistema
CREATE TABLE tb_funcionarios (
 CPF_funcionario VARCHAR(14) PRIMARY KEY,
 nome VARCHAR(100) NOT NULL,
 email VARCHAR(255) NOT NULL,
 senha VARCHAR(255) NOT NULL,
```

```
 SN VARCHAR(9) NOT NULL,

 foto VARCHAR(2048),

 permissao VARCHAR(30) NOT NULL,

 id_funcao INT,

 CONSTRAINT FK_funcionarios_funcao FOREIGN KEY (id_funcao) REFERENCES tb_funcoes
(id_funcao)

);


#Criar a tabela que armazena as solicitações de serviço

CREATE TABLE tb_solicitacoes (

 id_solicitacao INT PRIMARY KEY AUTO_INCREMENT,

 id_servico INT NOT NULL,

 id_sala VARCHAR(10) NOT NULL,

 descricao VARCHAR(2048) NOT NULL,

 CPF_funcionario VARCHAR(14) NOT NULL,

 foto VARCHAR(2048),

 data_inicio DATETIME NOT NULL,

 CONSTRAINT FK_tb_solicitacoes_servicos FOREIGN KEY (id_servico) REFERENCES
tb_servicos (id_servico),

 CONSTRAINT FK_tb_solicitacoes_sala FOREIGN KEY (id_sala) REFERENCES tb_salas (id_sala),

 CONSTRAINT FK_tb_solicitacoes_funcionario FOREIGN KEY (CPF_funcionario) REFERENCES
tb_funcionarios (CPF_funcionario)

);


#Criar a tabela que armazena os serviços a serem realizados pelos técnicos

CREATE TABLE tb_encaminhamentos (
```

```
id_encaminhamento INT PRIMARY KEY AUTO_INCREMENT,

CPF_funcionario VARCHAR(14) NOT NULL,

id_solicitacao INT NOT NULL,

urgencia VARCHAR(5) NOT NULL,

status VARCHAR(12) NOT NULL,

status_final VARCHAR(255),

adendo VARCHAR(2048),

status_finalizacao VARCHAR(255),

data_inicio_encaminhamento DATETIME NOT NULL,

data_inicio_servico DATETIME DEFAULT NULL,

data_termino_servico DATETIME DEFAULT NULL,

CONSTRAINT FK_tb_encaminhamento_funcionario FOREIGN KEY (CPF_funcionario)
REFERENCES tb_funcionarios (CPF_funcionario),

CONSTRAINT FK_tb_encaminhamento_solicitacao FOREIGN KEY (id_solicitacao)
REFERENCES tb_solicitacoes (id_solicitacao)

);


#Criar a tabela que armazena os relatórios finais

CREATE TABLE tb_relatorios (

id_relatorio INT PRIMARY KEY AUTO_INCREMENT,

id_encaminhamento INT NOT NULL,

data_relatorio DATETIME NOT NULL,

CONSTRAINT FK_tb_os_encaminhamento FOREIGN KEY (id_encaminhamento) REFERENCES
tb_encaminhamentos (id_encaminhamento)

);
```

```
INSERT INTO tb_funcoes (nome) VALUES ("Técnico de Manutenção"), ("Instrutor"),
("Secretaria"), ("OPP"), ("Coordenador"), ("Diretor"), ("Outros");

INSERT INTO tb_servicos (nome) VALUES ("Elétrica"), ("Hidráulica"), ("Pintura"),
("Alvenaria"), ("Maquinas");


SELECT * FROM tb_funcoes;

SELECT * FROM tb_funcionarios;

SELECT * FROM tb_servicos;

SELECT * FROM tb_salas;

SELECT * FROM tb_solicitacoes;

SELECT * FROM tb_encaminhamentos;

SELECT * FROM tb_relatorios;
```

Apêndice S – Códigos Adicionais

A configuração do CSS global define a normalização dos estilos, removendo margens e preenchimentos padrões e ajustando o tamanho base da fonte para 10px. Também aplica a fonte "Montserrat" em todo o corpo do site e define variáveis de cores personalizadas.

```
@import
url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0
,100..900;1,100..900&display=swap');


/* Normalização */

*{

    padding: 0;

    margin: 0;

    box-sizing: border-box;
```

```
}
/* Define o tamanho da fonte base */
html {
    font-size: 62.5%; /* 62.5% de 16px (padrão) é igual a 10px */

}

body {
    margin: 0;
    padding: 0;
    font-family: "Montserrat", sans-serif;
}

/* Váriaveis de Cor do Site*/
:root {
    --color01: rgb(255, 255, 255);
    --color02: rgb(0, 0, 0);
    --color03: #FFCD11;
    --color04: #F0F0F0;
    --color05: #ED7203;
    --color06: #2C2E41;
    --colorRed:#C61606;
    --colorGreen:#04A61C;
  }
```

O código JS contém uma função JavaScript chamada funcVoltar(), que é responsável por fazer o usuário retornar para a tela anterior na navegação do navegador.

```
// Função que faz o usuário voltar para a tela anterior
function funcVoltar() {
    window.history.back();
}
```

O código JS define uma função que gera uma saudação personalizada com base na hora do dia: "Bom dia" entre 6h e 12h, "Boa tarde" entre 12h e 18h, e "Boa noite" após as 18h. Em seguida, ele atualiza o conteúdo de um elemento HTML com o ID saudacao para exibir a mensagem apropriada.

```
// Função que gera uma saudação diferente de acordo com o horário
function obterSaudacao() {
    const agora = new Date();
    const hora = agora.getHours();

    if (hora >= 6 && hora < 12) {
        return "Bom dia, ";
    } else if (hora >= 12 && hora < 18) {
        return "Boa tarde, ";
    } else {
        return "Boa noite, ";
    }
}
```

```
document.getElementById('saudacao').textContent = obterSaudacao();
```

Este código define uma classe Connection com uma função conectar() que estabelece uma conexão com um banco de dados MySQL hospedado no Azure. A função utiliza o módulo mysql.connector e conecta-se ao banco de dados "bd_easyrequest" usando as credenciais especificadas (host, porta, usuário, senha e nome do banco). A conexão é retornada como um objeto que pode ser usado para interagir com o banco de dados.

```
# Importando o mysql.connector
import mysql.connector


# Classe que inicia a conexão com o Banco de Dados
class Connection:
    def conectar(): # Função que conecta o servidor no Banco de
Dados armazenado no Azure
        myBD = mysql.connector.connect(
            host = "bd-easy-request.mysql.database.azure.com",
            port = "3306",
            user="equipe_easyrequest",
            password="4_b4t@t45_3_m31@",
            database="bd_easyrequest"
        )
```

```
        return myBD
```