Versão 2.8.2-betA

10 DE março DE 2025



kleber pereira de almeida

PREFEITURA DE ITAPOÁ

RUA MARIANA MICHELS BORGES, 201 – ITAPEMA DO NORTE – CEP: 89.360-730 – iTAPOÁ/ SC

# DOCUMENTAÇÃO

## CENÁRIO ATUAL E *STAKEHOLDERS*

No cenário atual os atendimentos técnicos são realizados com base em uma planilha compartilhada via *Google Drive* entre os membros do departamento, a mesma é alimentada após atendimento telefônico e recentemente via mensagens de *WhatsApp*.

Os dados preenchidos a cada atendimento são:

\_ Id (único);  
\_ Status (solucionado, especializada, interno – em espera, em andamento, aguardando peças, prateleira, recomenda baixa, externo – em espera, sem solução, manutenção agendada, entregar equipamento, duplicado);  
\_ Secretaria/ Setor (administração, agricultura e pesca...);  
\_ Responsável (nome do profissional que solicitou o atendimento);  
\_ Problema/ Descrição (relatado pelo usuário);  
\_ Técnico (nome do profissional responsável por atender o chamado);  
\_ Data da Abertura;  
\_ Data do Atendimento;  
\_ Data do Encerramento;  
\_ Contato (telefone, *WhatsApp*, *e-mail*);  
\_ Solução Empregada (descrição detalhada do que foi realizado);  
\_ Patrimônio (registro único do bem patrimonial do município).  
  
As informações que são obtidas através de relatoria são:

\_ Quantidade de chamados em andamento;  
\_ Quantidade de chamados em espera;  
\_ Quantidade de chamados solucionados;  
\_ Quantidade de chamados aguardando peças;  
\_ Quantidade de chamados na prateleira;  
\_ Quantidade de chamados aguardando atendimento de especializadas (terceiros);  
\_ Quantidade de chamados total em abertos;  
\_ Quantidade de chamados sem solução;  
\_ Quantidade de chamados com manutenção agendada;  
\_ Quantidade média de duração para o atendimento dos chamados (contagem em dia);  
\_ Quantidade de chamados total;

Há ainda outra aba na planilha com estatísticas de desempenho, onde escolhe-se o ano desejado e automaticamente na tabela com nomes dos funcionários do departamento os valores da quantidade total de chamados atendidos e a taxa de atendimento (percentual correspondente) são atualizados.

### TABELA DOS *STAKEHOLDERS*

| *Stakeholders* | Nome | Responsabilidades |
| --- | --- | --- |
| Diretor do Departamento da Tecnologia da Informação | Rafael | \_ Verificar o andamento dos chamados; \_ Definir prioridades dos atendimentos; \_ Definir/ Aprovar projetos de implantação/ melhorias; \_ Aprovar compra de ferramentas; \_ Aprovar compra de *hardwares* e *softwares*; \_ Deslocar a equipe para os atendimentos externos; \_ Auxiliar nos trabalhos técnicos quando os mesmos exigem uma demanda maior; \_ Alimentar a planilha de chamados. |
| Coordenadora do Departamento da Tecnologia da Informação | Fernanda | \_ Realizar orçamentos e compras dos materiais necessários; \_ Realizar orçamentos e compras das ferramentas necessárias; \_ Verificar *WhatsApp* para abertura de chamados; \_ Definir prioridades dos atendimentos; \_ Deslocar a equipe para os atendimentos externos; \_ Solicitar atendimento de terceiros de acordo com suas contratações; \_ Supervisionar os serviços realizados por terceiros; \_ Auxiliar nos trabalhos técnicos quando os mesmos exigem uma demanda maior; \_ Supervisionar os trabalhos dos estagiários. \_ Alimentar a planilha de chamados. |
| Técnicos de Informática | Kleber | \_ Realizar manutenção dos equipamentos; \_ Atender telefonemas para abertura de chamados; \_ Verificar *WhatsApp* para abertura de chamados; \_ Alimentar a planilha de chamados; \_ Solicitar atendimento de terceiros de acordo com suas contratações; \_ Especificar recomendações técnicas; \_ Auxiliar nos trabalhos dos projetos, implantação e melhorias; \_ Supervisionar os serviços realizados por terceiros; \_ Supervisionar os trabalhos dos estagiários. |
| Estagiário do Departamento da Tecnologia da Informação | Jorge | \_ Realizar manutenção dos equipamentos; \_ Atender telefonemas para abertura de chamados; \_ Verificar *WhatsApp* para abertura de chamados; \_ Solicitar atendimento de terceiros de acordo com suas contratações; \_ Alimentar a planilha de chamados; |
| Funcionário de serviços gerais/ Contratado de serviços | Bilo/ Marielle | \_ Perfurar paredes; \_ Passar cabeamentos; \_ Fixar *racks* e outros móveis que exijam ferramentas e mão de obra específica.\_ |
| Terceirizado/ Contratado | -- | \_ Realizar os atendimentos das solicitações encaminhadas pelo Departamento da Tecnologia da Informação de acordo com sua contratação e prazos estipulados. |
| Demais Funcionários | -- | \_ Solicitar solução de problemas na área correlata via ligação telefônica, *WhatsApp*, *e-mail* e ou verbalmente; \_ Realizar etapas consideradas de baixo risco através de orientação técnica. |

## ENTREVISTA PARA A DEFINIÇÃO DO PROJETO

Para a definição do projeto, serão elencados alguns dos perfis dos *stakeholders*, afim de traçar as necessidades não atendidas pelo método atual.

### ENTREVISTADO - PERFIL DEMAIS FUNCIONÁRIOS

\_ Jairo Severino de Freitas;  
\_ Procuradoria Jurídica;  
\_ Agente Administrativo;

O entrevistado aceitou participar desse projeto respondendo às perguntas de acordo com seu ponto de vista sobre o assunto abordado, e desse momento extraiu-se a seguinte análise.

**Necessidade 1:** Alguns setores necessitam atender prazos judiciais que se não forem executados em tempo hábil se tornam irreversíveis no processo, nos casos mais críticos esse prazo é de até 24 horas.

**Análise 1:** O sistema deverá possuir um campo que identifique a prioridade do chamado, e as prioridades devem ser definidas por uma hierarquia, estando o usuário na base hierárquica podendo alterar a prioridade para um nível acima, afim de minimizar a demora nos atendimentos mais críticos.

**Necessidade 2: O acesso remoto é realizado através da aplicação *AnyDesk*, que é informado após ligação telefônica, uma alternativa mais ágil seria isso estar disponível no próprio sistema para agilizar o atendimento técnico.**

**Análise 2: O sistema deverá possuir um campo de preenchimento de código para o acesso remoto que será alimentado na abertura do chamado.**

### ENTREVISTADO - PERFIL ESTAGIÁRIO DO DEPARTAMENTO DE TECNOLOGIA DA INFORMAÇÃO

\_ Jorge Enrique Santos de Lima Takao Nishimura;  
\_ Departamento de Tecnologia da Informação;  
\_ Estagiário;

O entrevistado aceitou participar desse projeto respondendo às perguntas de acordo com seu ponto de vista sobre o assunto abordado, e desse momento extraiu-se a seguinte análise.

**Necessidade 3:** Para alimentar os dados referentes ao chamado alguns usuários não saberão onde localizá-los, i.e., tem gente que não sabe onde localizar o SELB da impressora (identificação das impressoras locadas).

**Análise 3:** O sistema deverá conter ícones para tirar dúvidas dos usuários referentes ao preenchimento do campo e a ajuda dependendo do caso deverá ter imagem ilustrativa auxiliando na localização da informação.

**Necessidade 4: Ao abrir o chamado o usuário poderia receber uma orientação prévia, afim de auxiliar na solução do problema.**

**Análise 4: O sistema deverá possuir uma análise heurística, um banco de soluções ou um campo extra onde o usuário relata as etapas que já foram executadas.**

### ENTREVISTADO - PERFIL COORDENADORA DO DEPARTAMENTO DA TECNOLOGIA DA INFORMAÇÃO

\_ Fernanda Luna;  
\_ Departamento de Tecnologia da Informação;  
\_ Coordenadora;

A entrevistada aceitou participar desse projeto respondendo às perguntas de acordo com seu ponto de vista sobre o assunto abordado, e desse momento extraiu-se a seguinte análise.

**Necessidade 5:** No método atual o usuário liga e o profissional do Departamento da Tecnologia da Informação alimenta os dados na planilha, porém, muitos desses dados não são preenchidos ou quando preenchidos são incompletos.

**Análise 5:** O sistema deve permitir que o profissional do Departamento de Tecnologia da Informação abra, altere, encerre o chamado inclusive em nome do solicitante, porém, o sistema deve registrar que o chamado foi aberto pelo profissional do departamento em nome do solicitante.

**Necessidade 6: Um ponto positivo que a planilha atende hoje é a de gerar o tempo de atendimento, principalmente dos serviços terceirizados, e a coordenação gostaria de manter e inclusive melhorar isso, determinando o tempo gasto também pela empresa prestadora do serviço.**

**Análise 6: O sistema deverá registrar automaticamente a data e hora de abertura de chamado, data e hora do início do atendimento e data e hora do encerramento do atendimento nos casos de atendimento direto, sempre encaminhando *e-mail* automaticamente ao solicitante informando em qual etapa encontra-se. Para os atendimentos indiretos (encaminhados a terceiros) ao encaminhar o serviço para terceiro o profissional do departamento deverá encerrar o atendimento direto, e o sistema gera um chamado dependente, constando o número de chamado com data e hora de abertura, data e hora de atendimento de acordo com o estipulado pelo contrato vigente e após decorrido o prazo limite o sistema deve automaticamente passar para a etapa de serviço executado gerando um *e-mail* para o solicitante informando do fim do prazo para o atendimento. Para os profissionais do Departamento de Tecnologia da Informação o sistema deverá emitir uma notificação indicando que o prazo limite para atendimento foi atendido e que não houve encerramento do mesmo.**

### ENTREVISTADO - PERFIL DIRETOR DO DEPARTAMENTO DA TECNOLOGIA DA INFORMAÇÃO

\_ Rafael Vida Almeida;  
\_ Departamento de Tecnologia da Informação;  
\_ Diretor;

O entrevistado aceitou participar desse projeto respondendo às perguntas de acordo com seu ponto de vista sobre o assunto abordado, e desse momento extraiu-se a seguinte análise.

**Necessidade 7:** No método atual qualquer um dos profissionais pode excluir um chamado, gostaria que no sistema proposto aqui isso não fosse possível.

**Análise 7:** O sistema deverá permitir apenas a inatividade do registro do chamado pelo profissional do Departamento de Tecnologia da Informação, afim de manter o histórico do mesmo.

**Necessidade 8: Gostaria que os relatórios da planilha fossem mais completos, ao ponto de permitir análise de peças mais trocadas, problemas persistentes e setores recorrentes.**

**Análise 8: O sistema deve gerar relatórios que demonstrem trocas de peças, problemas persistentes, setores com problemas recorrentes dentre outros.**

## REQUISITOS DO SISTEMA

### REGRAS DE NEGÓCIO

Definem as condições em que um processo é realizado ou as novas condições após a conclusão de um processo.

**[RN001]** Para acessar o sistema os usuários deverão utilizar de um dispositivo operante com sistema operacional, navegador, conexão com a Internet e dados de *login* devidamente registrados.

**[RN002]** Para abertura de chamados, os usuários deverão fornecer os dados obrigatórios.

**[RN003]** Para recuperar os dados de *login* os usuários deverão informar endereço de *e-mail* registrado, caso não se recorde do endereço de *e-mail* o usuário deverá contatar o administrador do sistema.

**[RN004]** Em casos onde os usuários não se recordarem do *e-mail* registrado para recuperação de senha e solicitarem ao administrador do sistema a alteração dos dados o mesmo deve se certificar de estar atendendo o proprietário pela conta em questão.

**[RN005]** Quando o sistema apresentar algum defeito, erro ou falha o administrador do sistema deverá solicitar reparo à equipe de desenvolvimento.

\_ **Defeito** é qualquer imperfeição ou inconsistência no produto do software ou em seu processo, um defeito é também uma não conformidade, ou seja, o defeito faz parte do produto, é algo que está implementada no código de maneira errada.  
\_ **Erro** pode ser um resultado de um defeito ou falha, como um retorno esperado, que por causa de uma falha teve um valor diferente do esperado.  
\_ **Falha** está mais ligada ao hardware, como uma rede inacessível ou queda de energia. Uma falha pode ocorrer por causa de um erro, por exemplo, houve um retorno de um valor não esperado, como *null*, isso é um erro, e por causa desse *null* ocasionou uma falha no sistema.

**[RN006]** Quando o administrador do sistema não se recordar de seu *e-mail* para recuperação de senha o mesmo deverá solicitar à equipe de desenvolvimento.

**[RN007]** O usuário deverá acompanhar o andamento do chamado por *e-mail* e/ ou acessando o sistema.

**[RN008]** Caso o usuário esteja impedido de abrir ou acompanhar o chamado pelo sistema, o mesmo deverá ligar ou encaminhar mensagem via *Whatsapp* para o Departamento da Tecnologia da Informação.

**[RN009]** Caso o usuário contate o Departamento de Tecnologia da Informação solicitando abertura de chamado ou acompanhamento do mesmo, os profissionais do departamento deverão abrir ou informar a situação o chamado.

**[RN010]** Os chamados redirecionados para terceirizados deverão ser alimentados em suas aplicações específicas.

**[RN011]** Os usuários deverão encerrar os chamados quando os mesmos julgarem que o atendimento não é mais necessário.

**[RN012]** Os usuários deverão ser capacitados para o uso da aplicação.

**[RN013]** O usuário deverá registrar uma nova senha que atenda o requisito do sistema constante no **[RF011]**.

**[RN014]** As solicitações de acesso ao sistema deverão ser aprovadas por perfil com privilégio acima do nível 2 (coordenador, diretor e administrador), e o responsável pela liberação deverá consultar os dados do solicitante junto ao sistema do portal do munícipio antes de validá-lo.

### REQUISITOS FUNCIONAIS

Os requisitos funcionais referem-se sobre o que o sistema deve fazer, ou seja, suas funções e informações, preocupam-se com a funcionalidade e os serviços do sistema, as funções que o sistema deve fornecer para o cliente e como o sistema se comportará em determinadas situações.

[**RF001]** O sistema deve registrar, ativar e inativar usuários.

**[RF002]** O sistema deve registrar, alterar e inativar os chamados.

**[RF003]** O usuário pode consultar seus dados no sistema.

**[RF004]** O sistema deve gerar relatórios gráficos dos chamados.

**~~[RF005]~~** ~~O sistema deve enviar~~ *~~e-mail~~* ~~para os usuários registrados em cada uma das etapas do chamado.~~

**[RF006]** O sistema deve permitir que o usuário escolha quais notificações quer receber via *e-mail.*

**[RF007]** O sistema deve permitir que o usuário recupere sua senha para acesso.

**[RF008]** O sistema deve enviar *e-mail* para recuperação de senha.

**[RF009]** O sistema deve permitir a inserção de imagens de no máximo 2 por etapa, cada imagem deverá ter o tamanho de no máximo 700MB.

**~~[RF010]~~** ~~O sistema deve permitir o registro de agendamento de atualizações, contendo ambiente de execução, tipos de atualização e descrição da atualização.~~

**[RF011]** Ao registrar um novo usuário o sistema deverá gerar uma senha aleatória com os seguintes requisitos, de 7 a 14 caracteres, devendo conter ao menos uma letra e ao menos um número, não sendo aceito caracteres especiais.

**[RF012]** Ao registrar um novo usuário o sistema deverá enviar *e-mail* com link de alteração de senha.

**[RF013]** Caso no registro de um dispositivo seja preenchido o campo “patrimônio/ selb”, o sistema não deve permitir a inserção de duplicados.

**[RF014]** O sistema deve conter um registro de equipamento indefinido e o mesmo não poderá ter seus atributos alterados.

**~~[RF015]~~** ~~O sistema deve permitir que usuário com privilégios de administrador do sistema adicione informações em qualquer campo através do menu “abrir chamado”.~~

### REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais referem-se aos critérios que qualificam os requisitos funcionais, são geralmente mensuráveis e assim devemos preferencialmente associar uma medida ou referência para cada requisito não funcional.

**[RNF001]** As linguagens utilizadas para o desenvolvimento da aplicação serão *HTML, Javascript, CSS, PHP, MySQL.*

**[RNF002]** Os *frameworks*, pacotes e bibliotecas utilizadas serão, *Composer, AdminLTE3 e Mpdf.*

**[RNF003]** A metodologia aplicada nas linguagens deverá ser Procedural e/ou Orientada a Objeto.

**[RNF004]** As nomenclaturas dos botões devem conter verbos, afim de expressar a ação desejada (acessar, recuperar...).

**[RNF005]** As consultas do sistema não devem ultrapassar o tempo de 8 segundos.

**[RNF006]** As variáveis, funções, métodos e demais rotinas do sistema devem seguir o padrão *camelCase* e em português do Brasil, sem abreviaturas, salvo em casos onde o anglicismo ou abreviaturas são aplicáveis (*pdf, usb, upa e etc...*) ou referenciam atributos e/ ou elementos da linguagem HTML desde que não conflitam com a documentação da linguagem PHP. Quando utilizadas em estruturas de repetição priorizar a letra “i” para o quantitativo primário (*for( $i = 0; $i <= 10; $i++)*), e a letra “j” para o quantitativo secundário, por exemplo, em casos com matrizes. Para os demais quantitativos sugere-se o uso da letra “n” ou a que melhor atender ao algoritmo.

**[RNF007]** O sistema deverá ter suas divisões de diretórios por módulos.

**[RNF008]** A estrutura do sistema deverá seguir uma metodologia *MVC.* Sugere-se um diretório “App” e em seu subnível, diretório “modelos” (*Models*), diretório “telas” (*Views*) e diretório “sistema” (*Controllers*).

**[RNF009]** Cada bloco do algoritmo ou função deverá ser comentado afim de esclarecer dúvidas futuras, demarcando seu início e fim, com descrição no seu início.

**[RNF010]** Deverá realizar *backup* do banco de dados diariamente e sempre que for realizar alguma manutenção do sistema.

**[RNF011]** Deverá realizar *backup* do sistema sempre que for efetuar alguma manutenção no sistema.

**[RNF012]** Sempre que for realizada manutenção do sistema, o acesso ao mesmo deverá ser restrito à equipe de desenvolvimento.

**[RNF013]** As imagens anexadas ao sistema não poderão ultrapassar o tamanho de 600KB.

**[RNF014]** As nomenclaturas dos arquivos das páginas deverão ser escritas no padrão *camelCase* em português do Brasil sem caracteres especiais (´, ~, ”, ç, &...), no substantivo (inicio.php) ou imperativo afirmativo da segunda pessoa do singular podendo ser seguido de um adjetivo (alteraDados.php).

**[RNF015]** As nomenclaturas das pastas deverão ser escritas em minúsculas em português do Brasil sem caracteres especiais (´, ~, ”, ç, &...), no substantivo (telas/).

**[RNF016]** O versionamento do sistema deverá ser determinado pelo seguinte padrão “versão 3.12-dev”.  
\_ Numeração do versionamento: quantidade de módulos do sistema, ex.: 3.X.X-XXX / última versão do BD, ex.: X.8.0-*XXX*;  
\_ Ambiente de execução: Em desenvolvimento = *dev*/ Em uso com alterações inferiores a 90 dias = beta/ Em uso com alteração superior a 90 dias = *stable* ex.: X.X.X-dev;

**[RNF017]** Os dados definidos com nível de segurança maior deverão ser criptografados em *hash* *SHA256*, tais como senhas ou dados do usuário via parâmetro *get*.

**[RNF018]** A estrutura de menu do sistema deverá receber identificadores para configuração de sua nomenclatura e quando o mesmo necessitar de espaço entre os numerais, deverá usar *underscore* (menu1.php, menu1\_2.php).

**[RNF019]** O sistema deverá emitir uma notificação na tela Inicial sempre que tiver uma manutenção agendada.

**[RNF020]** O sistema deverá permitir que o usuário escolha sua prioridade na abertura do chamado.

**[RNF021]** O sistema deverá permitir que o técnico, ou coordenador, ou diretor do Departamento de Tecnologia da Informação redefina a prioridade.

**[RNF022]** Para desempate de ordem de atendimento serão atribuídas as seguintes regras:  
\_ Diretor do DTI = Peso 4;  
\_ Coordenador do DTI = Peso 3;  
\_ Técnico e Administrativo do DTI = Peso 2;  
\_ Usuário = Normal e Alta com Peso 1;  
\_ Sistema = Ordem de abertura (Normal com Peso 0);

**[RNF023]** Caso a prioridade tenha sido alterada por um perfil de permissão mais alta, o sistema não poderá permitir que perfis de permissão mais baixa altere a prioridade;

**[RNF024]** O sistema deverá gerar o protocolo da solicitação de suporte de forma única e incremental;

**[RNF025]** O sistema deverá exibir as solicitações com cores de acordo com sua prioridade;

**[RNF026]** Sugere-se adotar as cores do protocolo de Manchester:  
\_ Peso 5 - Emergência = Vermelha;  
\_ Peso 4 - Muito Urgente = Laranja;  
\_ Peso 3 - Urgente = Amarela;  
\_ Peso 2 - Alta = Verde;  
\_ Peso 1 - Normal = Azul;

**[RNF027]** Os campos do tipo “*hidden*” deverão ser declarados antes do botão “*submit*” do formulário;

**[RNF028]** O código fonte deverá conter área de testes comentada com QA – área para testes, afim de facilitar as análises futuras;

**[RNF029]** O número de protocolo deverá conter o ano da abertura, seguido de seu ID, totalizando uma sequência de 9 caracteres, sendo completados com zeros à esquerda ex.: 202400001.

## TECNOLOGIAS EMPREGADAS

### FRONT END

|  |  |  |
| --- | --- | --- |
| AdminLTE 3 | Tema de painel de controle responsivo | https://adminlte.io/ |

### BACK END

|  |  |  |
| --- | --- | --- |
| 10.4.28-MariaDB | BD |  |

### DESENVOLVIMENTO

|  |  |  |
| --- | --- | --- |
| NetBeans IDE 19 | IDE | https://netbeans.apache.org/front/main/ |
| Xampp 3.3.0 | Ambiente de desenvolvimento para testes | https://www.apachefriends.org/pt\_br/index.html |
| Composer 2.6.5 | Gerenciamento de bibliotecas | https://getcomposer.org/ |
| Composer 2.7.x-dev | Autoload | "autoload": {  "psr-4": {  "App\\": "App/"  }  } |
| MySQL Wrokbench 8.0 CE | Modelagem de Banco de Dados | https://www.mysql.com/products/workbench/ |
| GitHub | Colaboração de Projeto | https://github.com/kleberjlle/sspmi/ |
| FileZilla | Envio FTP |  |

## VALIDAÇÃO DO PROTÓTIPO DE ALTA FIDELIDADE

### APONTAMENTOS

25/03/2024 a 26/03/2024:  
\_ Necessidade 1: Atendida;  
\_ Necessidade 2: Atendida;  
\_ Necessidade 3: Atendida parcialmente com sugestão de implantação de legenda no campo prioridade, afim de orientar o usuário sobre quais *tickets* serão considerados prioridade alta.  
\_ Necessidade 4: Não será atendida na primeira entrega, porém está definida no escopo para entrega posterior.  
\_Necessidade 5: Atendida com alteração, não será possível reabrir um chamado, afim de garantir a análise heurística futura.  
\_ Necessidade 6: Atendida.  
\_ Necessidade 7: Atendida.  
\_ Necessidade 8: Não será atendida na primeira entrega, porém está definida no escopo para entrega posterior.

Alterações:  
\_Os tickets deverão aparecer para todos os funcionários do mesmo setor, informando sempre na abertura do mesmo de que já há solicitações em aberto.  
\_ Implantar área de print para os técnicos anexarem imagens, afim de complementar os atendimentos técnicos.  
\_ Desabilitar o campo inativar para o próprio usuário.

Participantes:  
\_ Rafael Vida Almeida.  
\_ Fernanda Luna.  
\_ Sulmária Maria da Silva.  
\_ Jairo Severino de Freitas.  
\_ Lucas Rudnik.  
\_ Kevin Gulla.  
\_ Kleber Pereira de Almeida.

### PROTÓTIPO DE ALTA FIDELIDADE

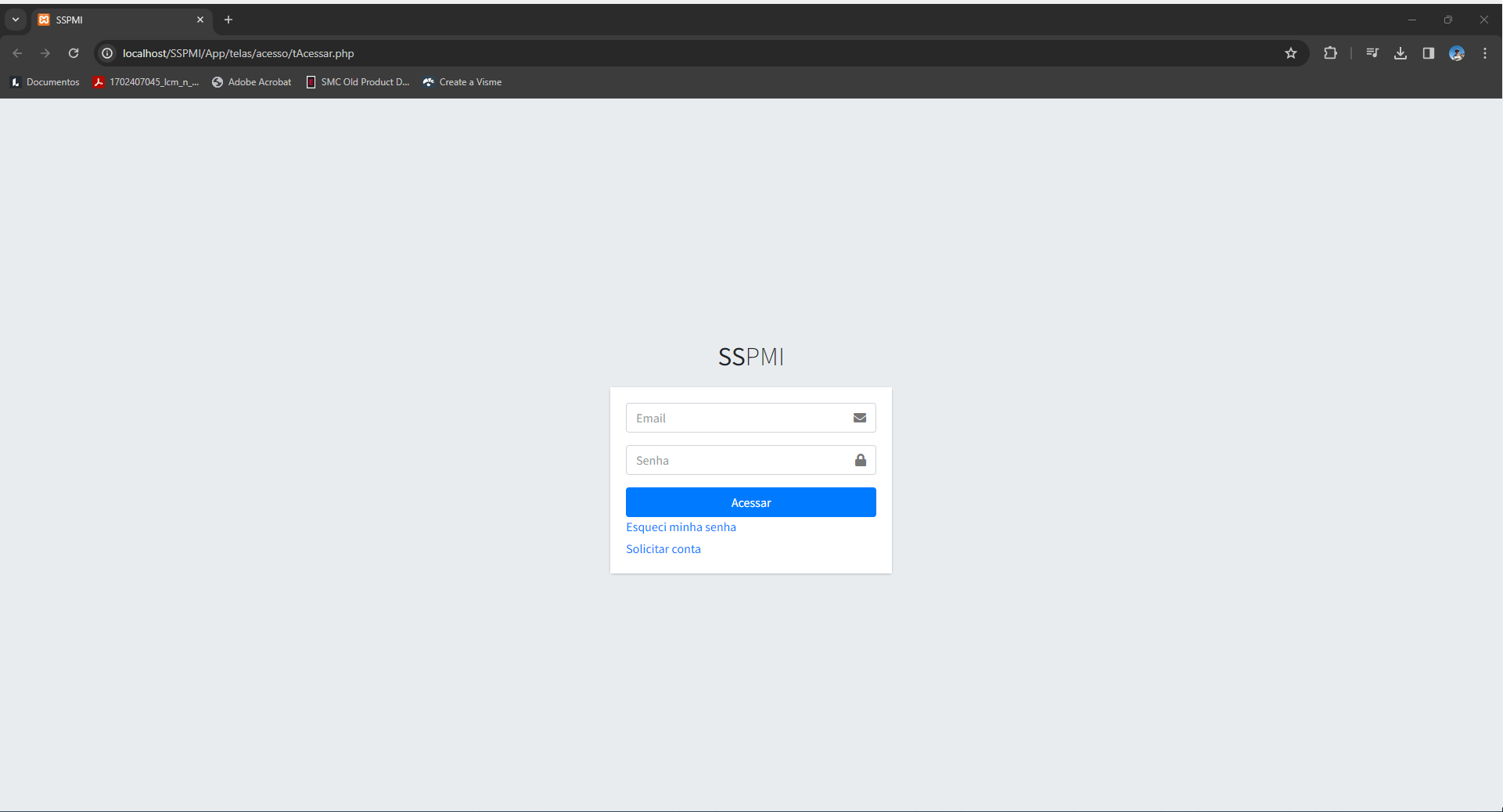


Figura - tAcessar.php

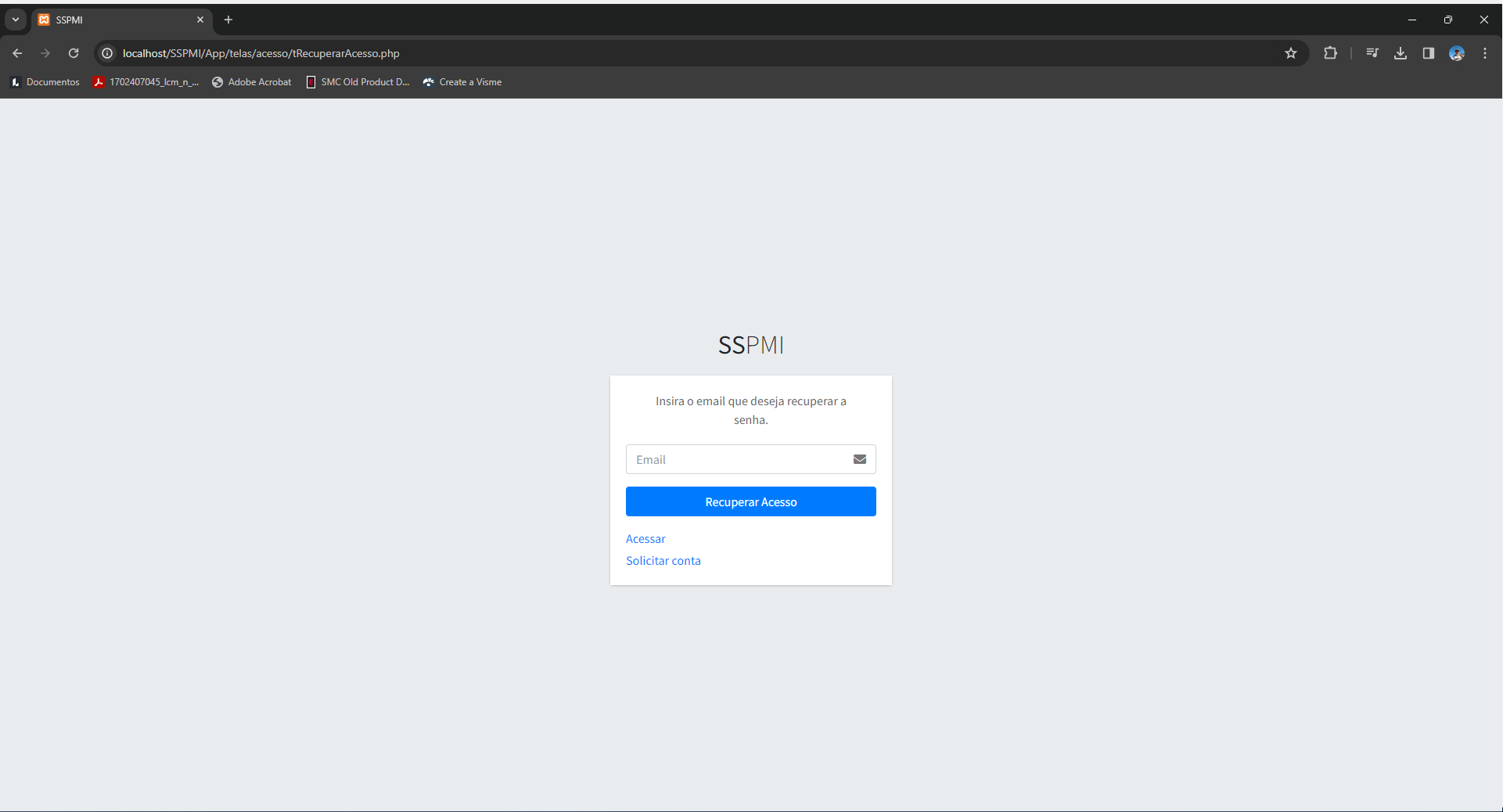


Figura - tRecuperaAcesso.php

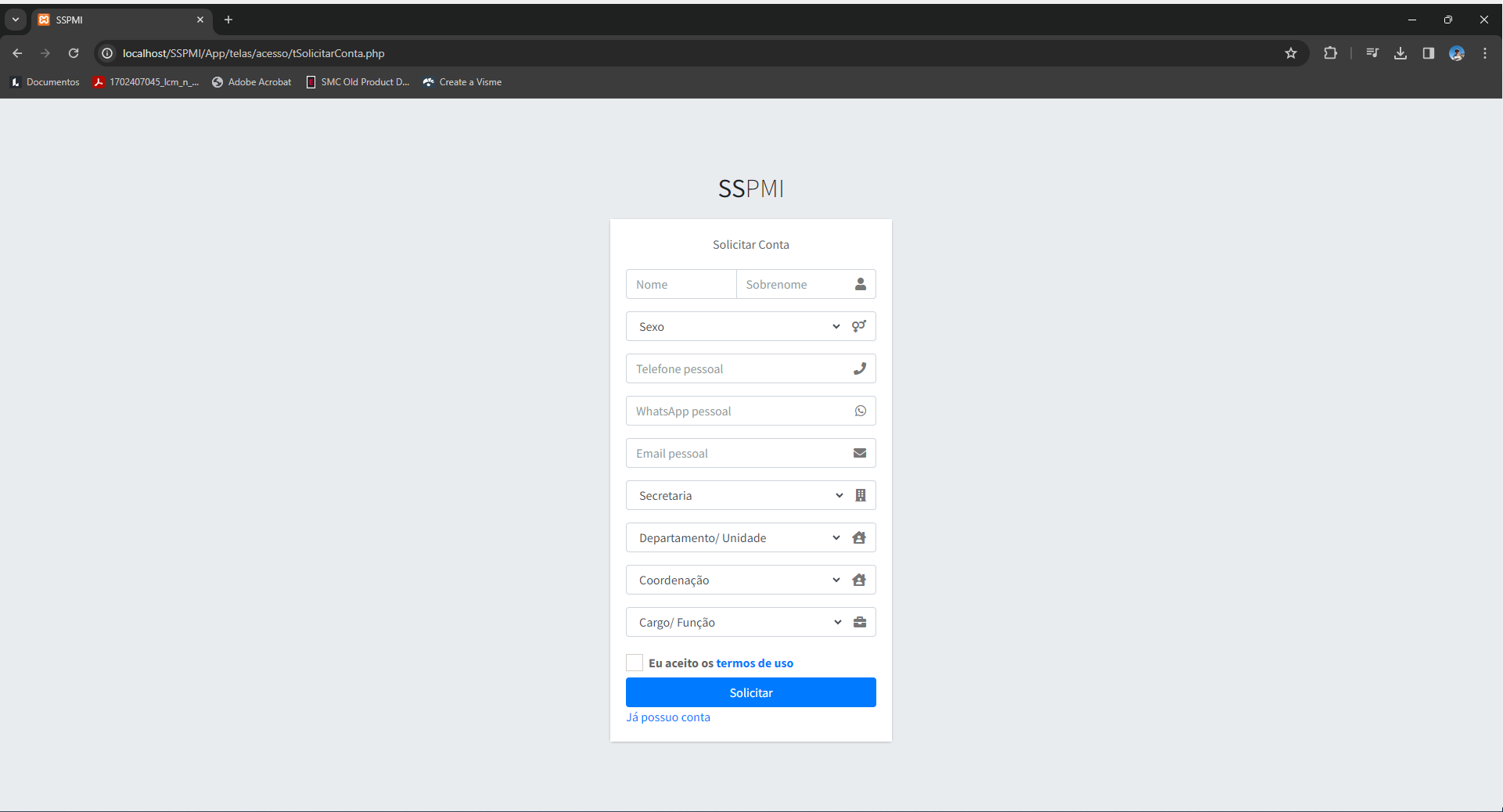


Figura - tSolicitaAcesso.php

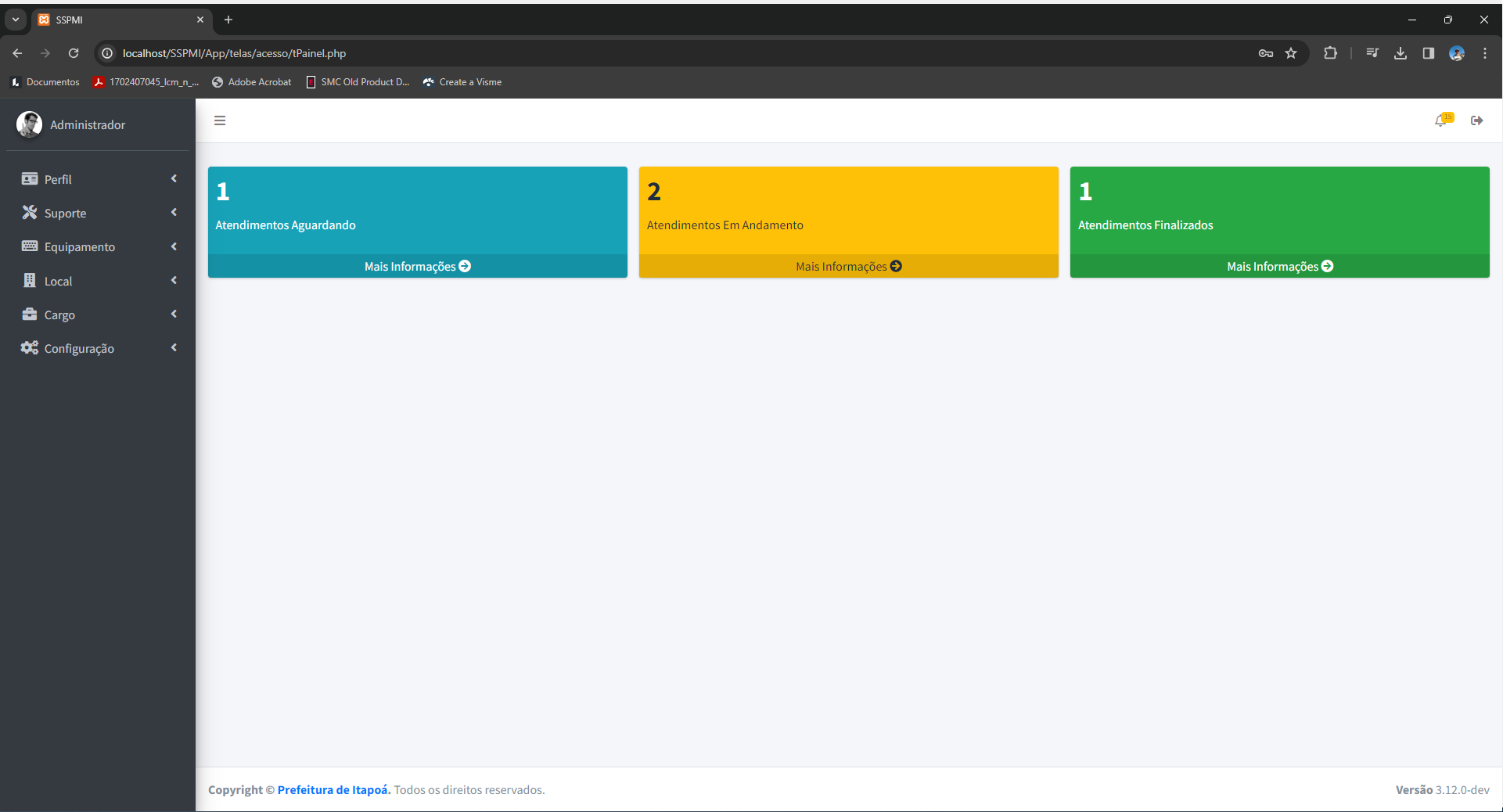


Figura - tPainel.php

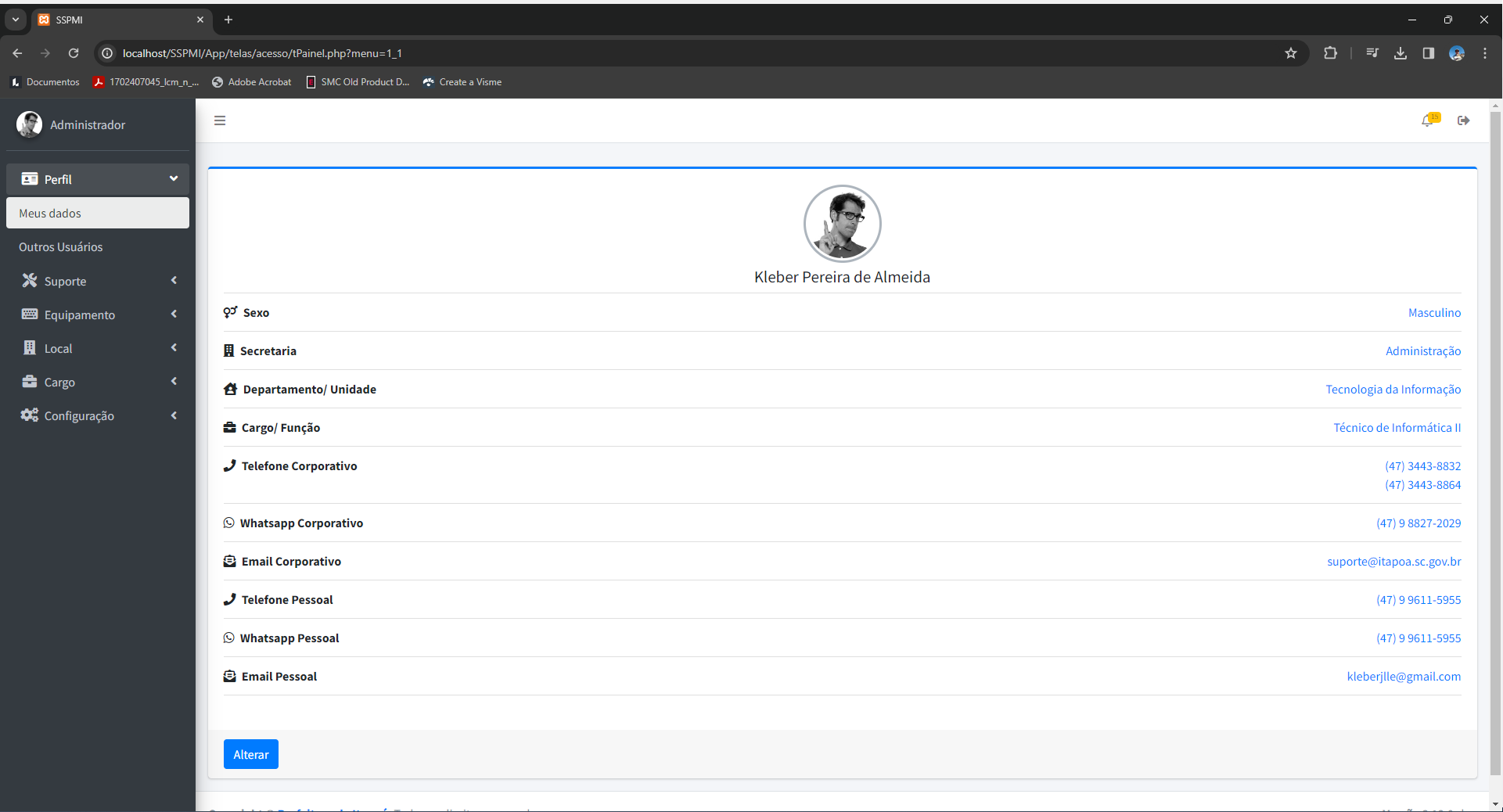


Figura - menu1\_1.php

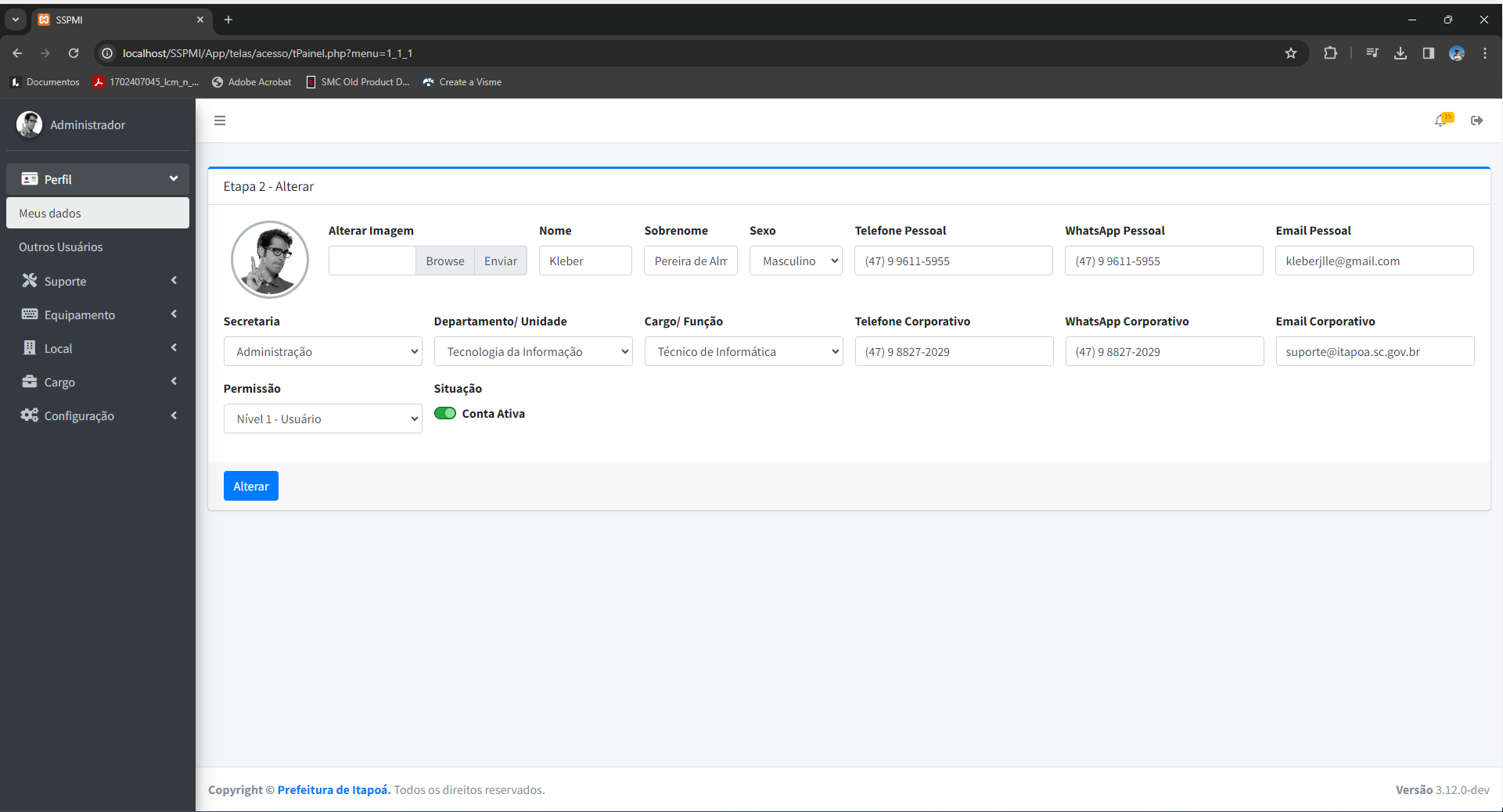


Figura - menu1\_1\_1.php

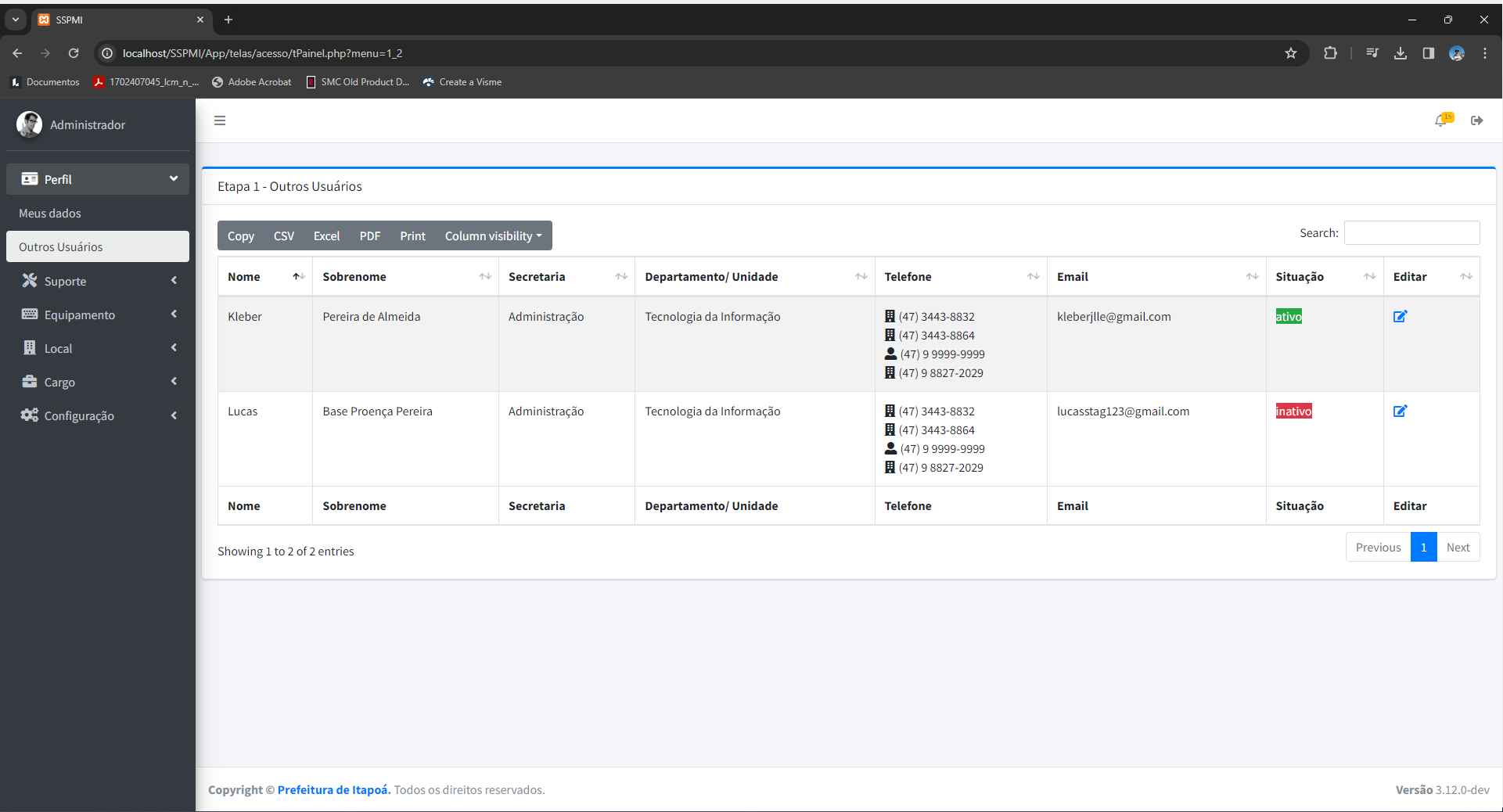


Figura - menu1\_2.php

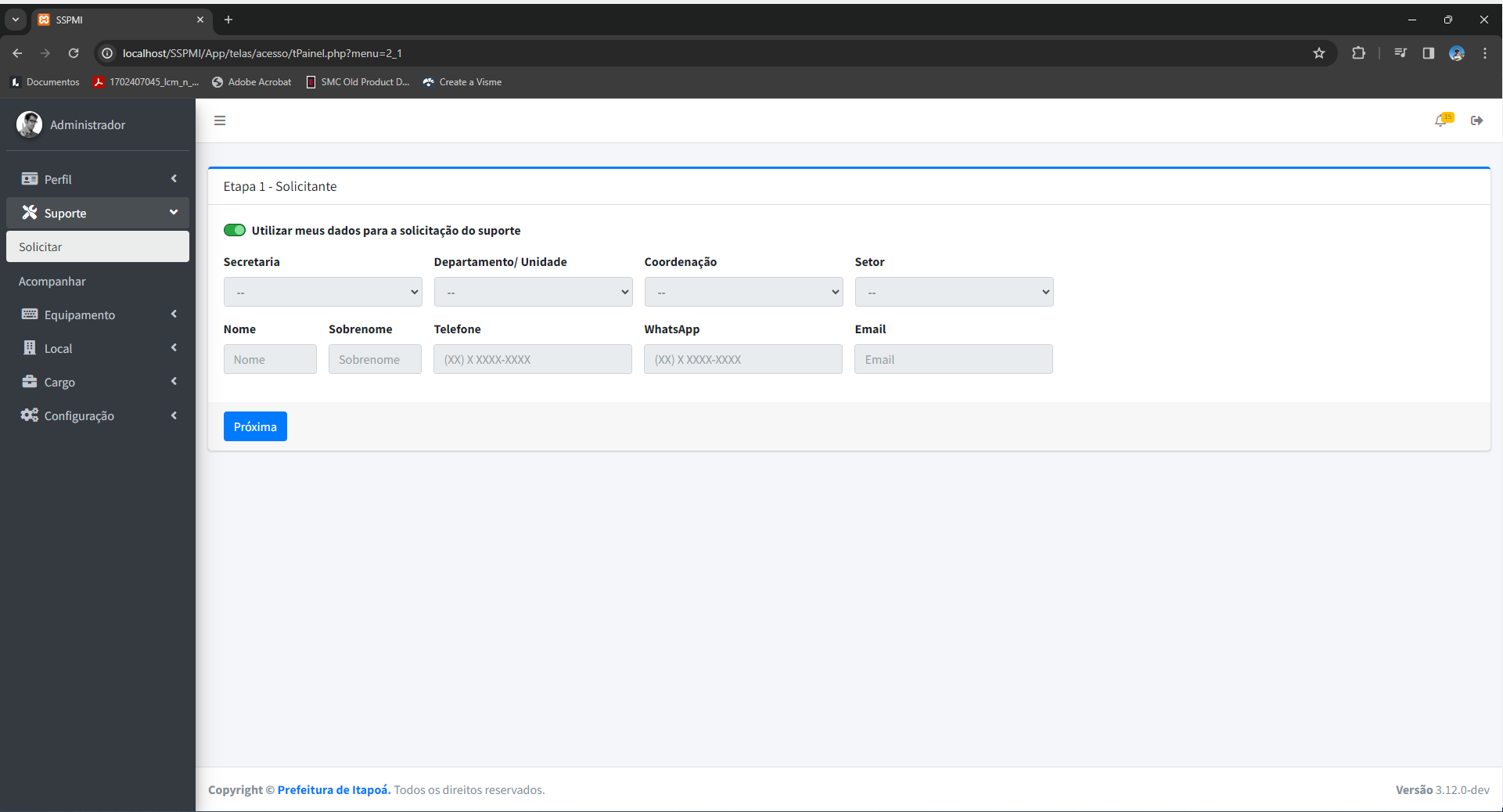


Figura - menu2\_1.php

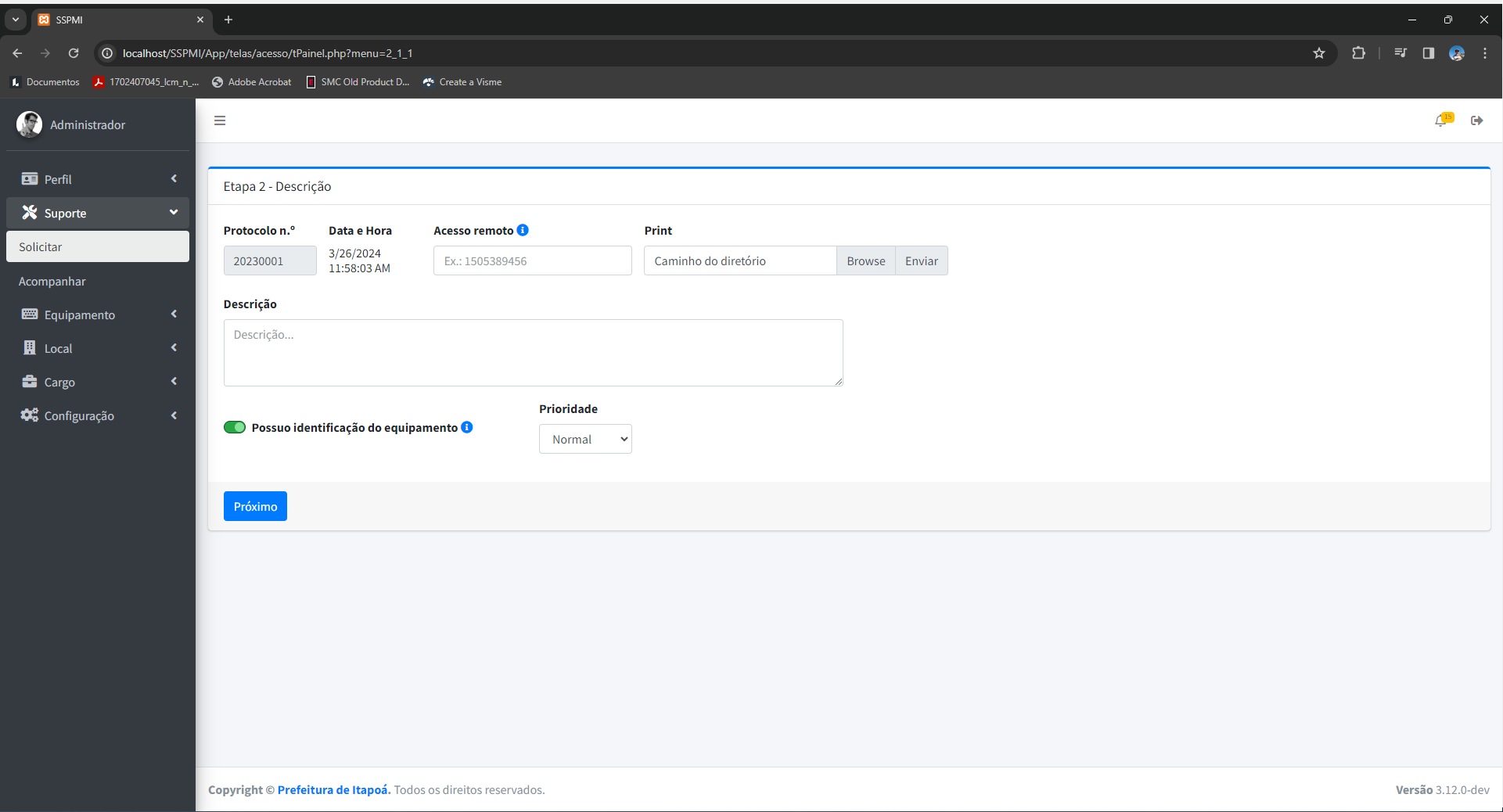


Figura - menu2\_1\_1.php

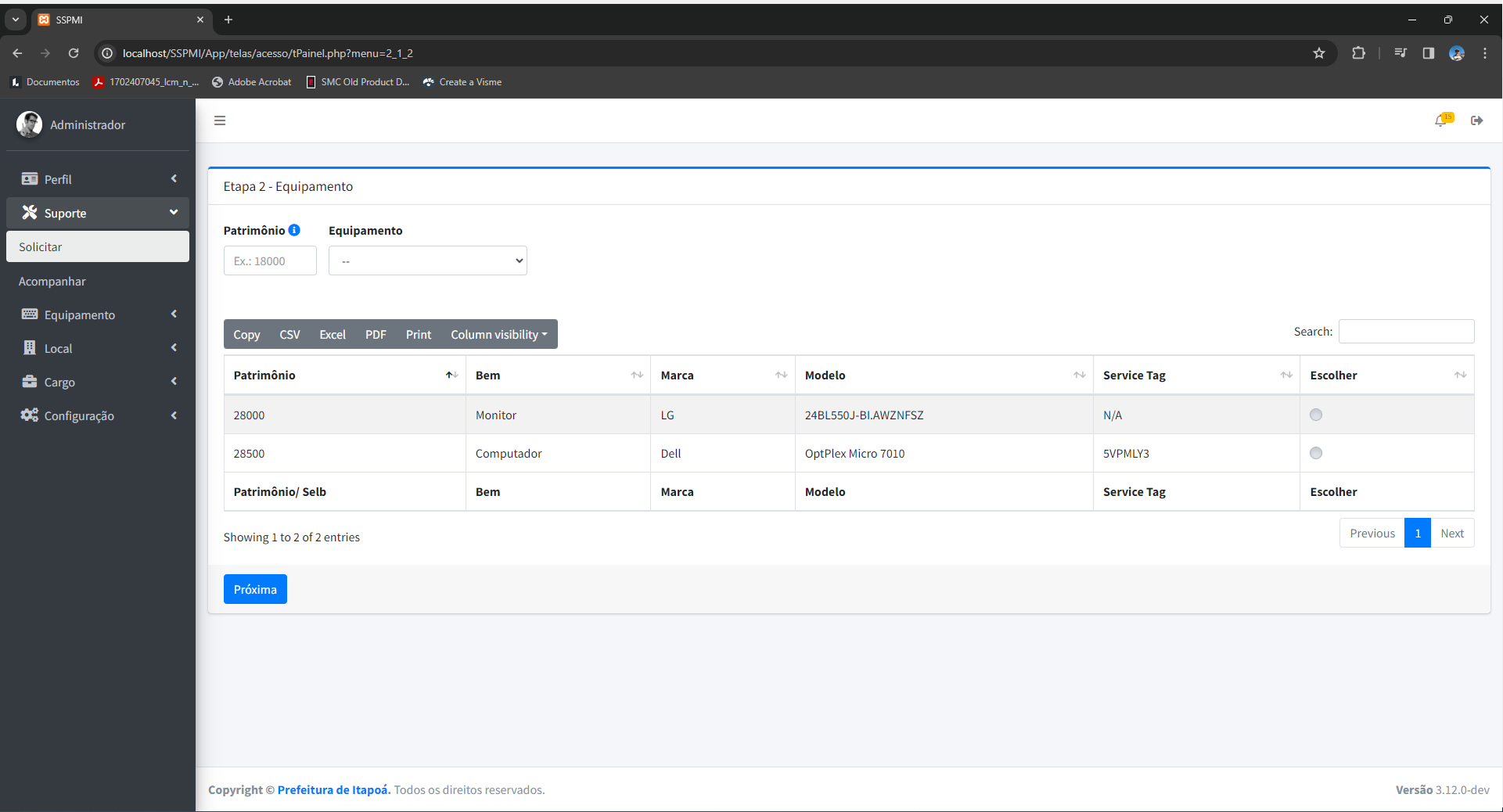


Figura - menu2\_1\_2.php

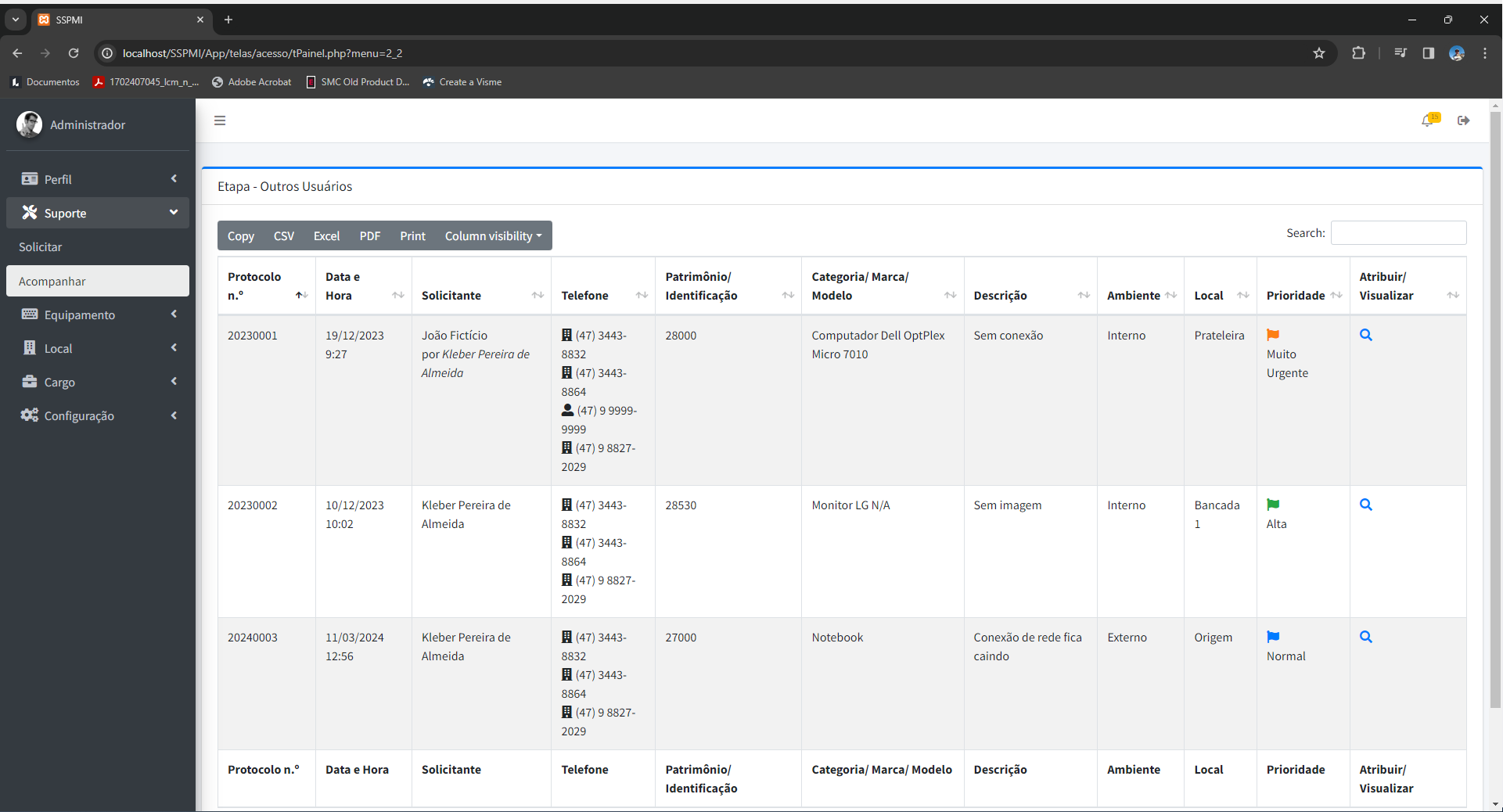


Figura - menu2\_2.php

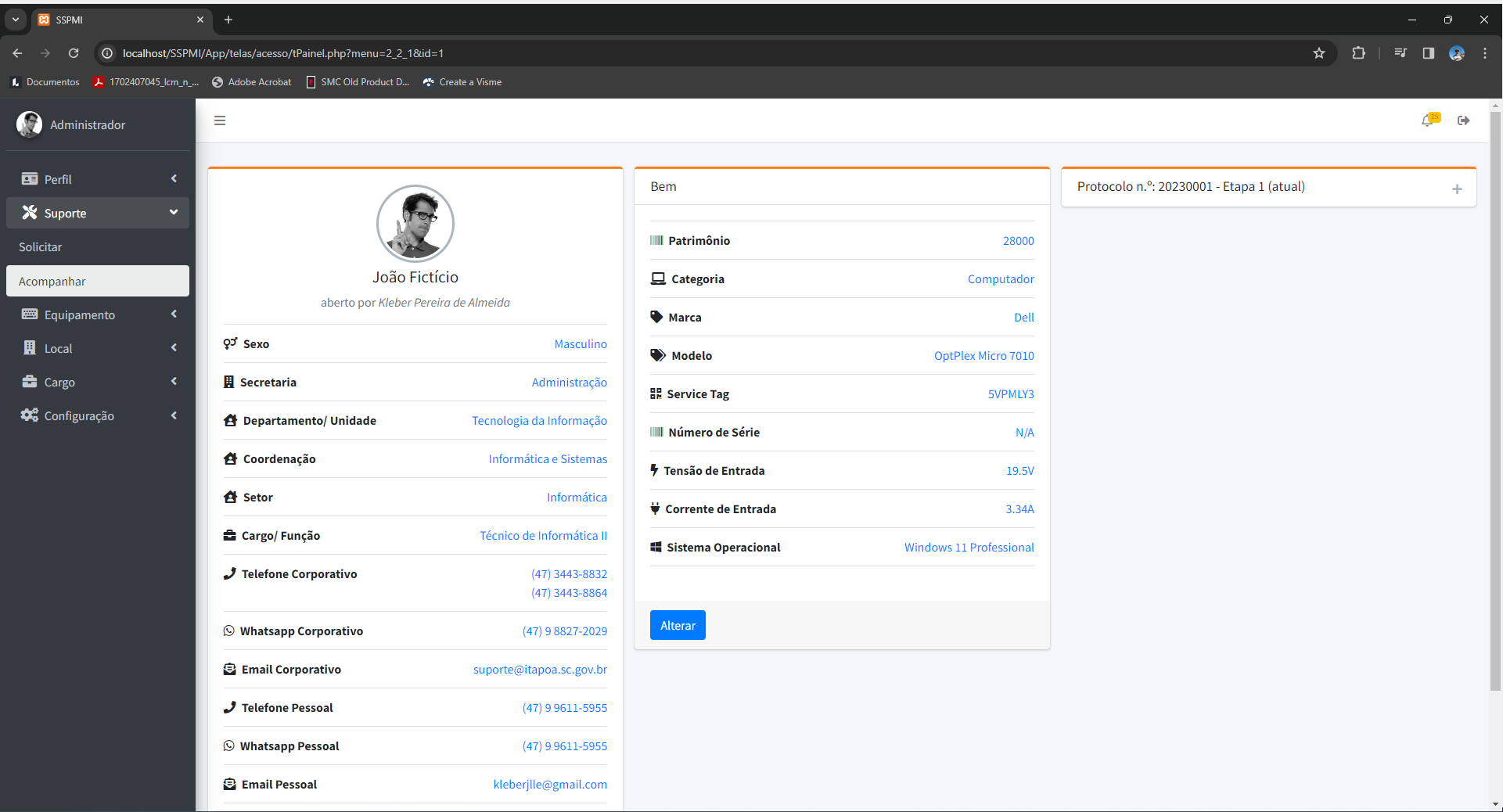


Figura - menu2\_2\_1.php

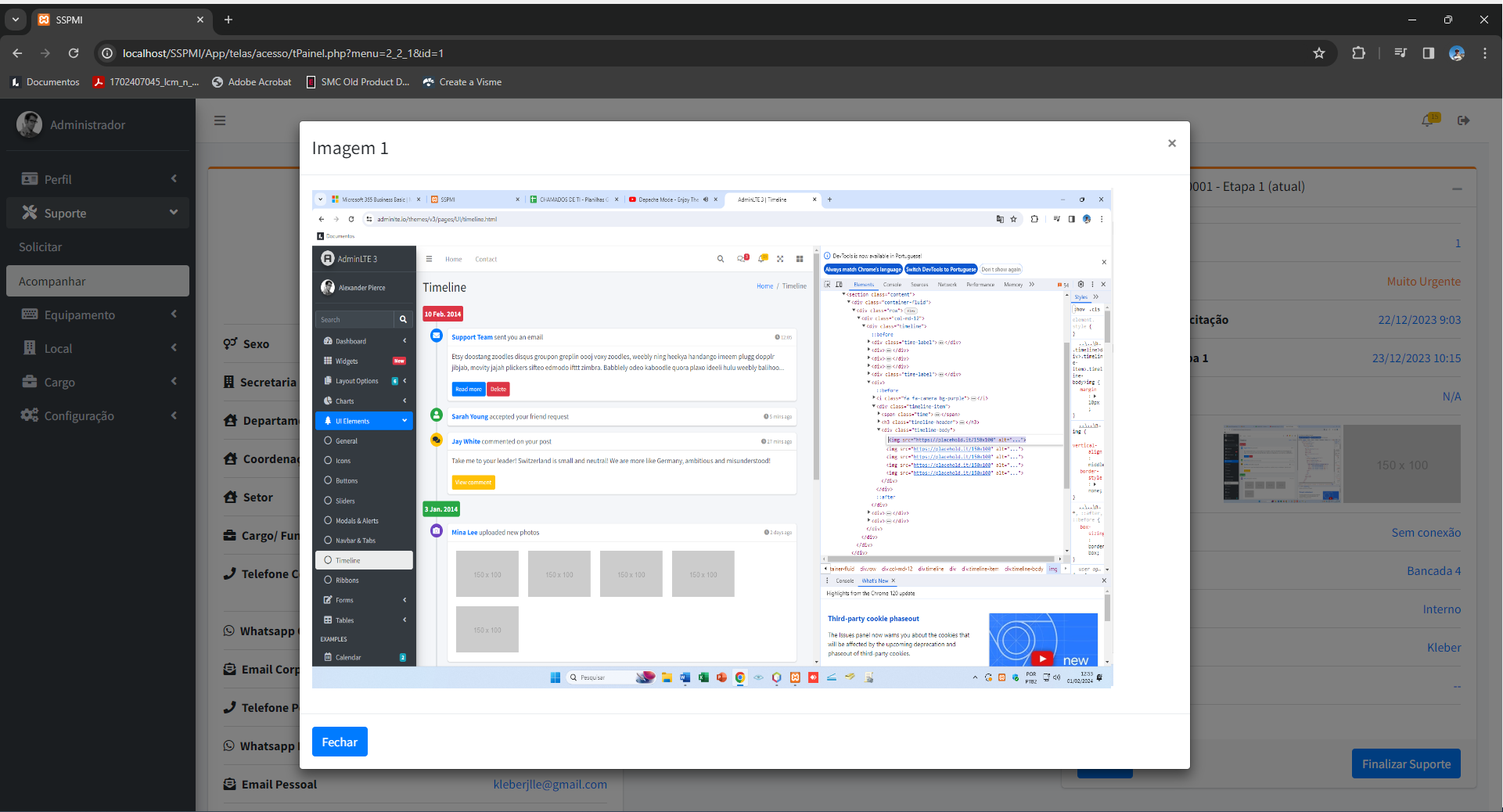


Figura - Modal print1 do menu2\_2\_1.php

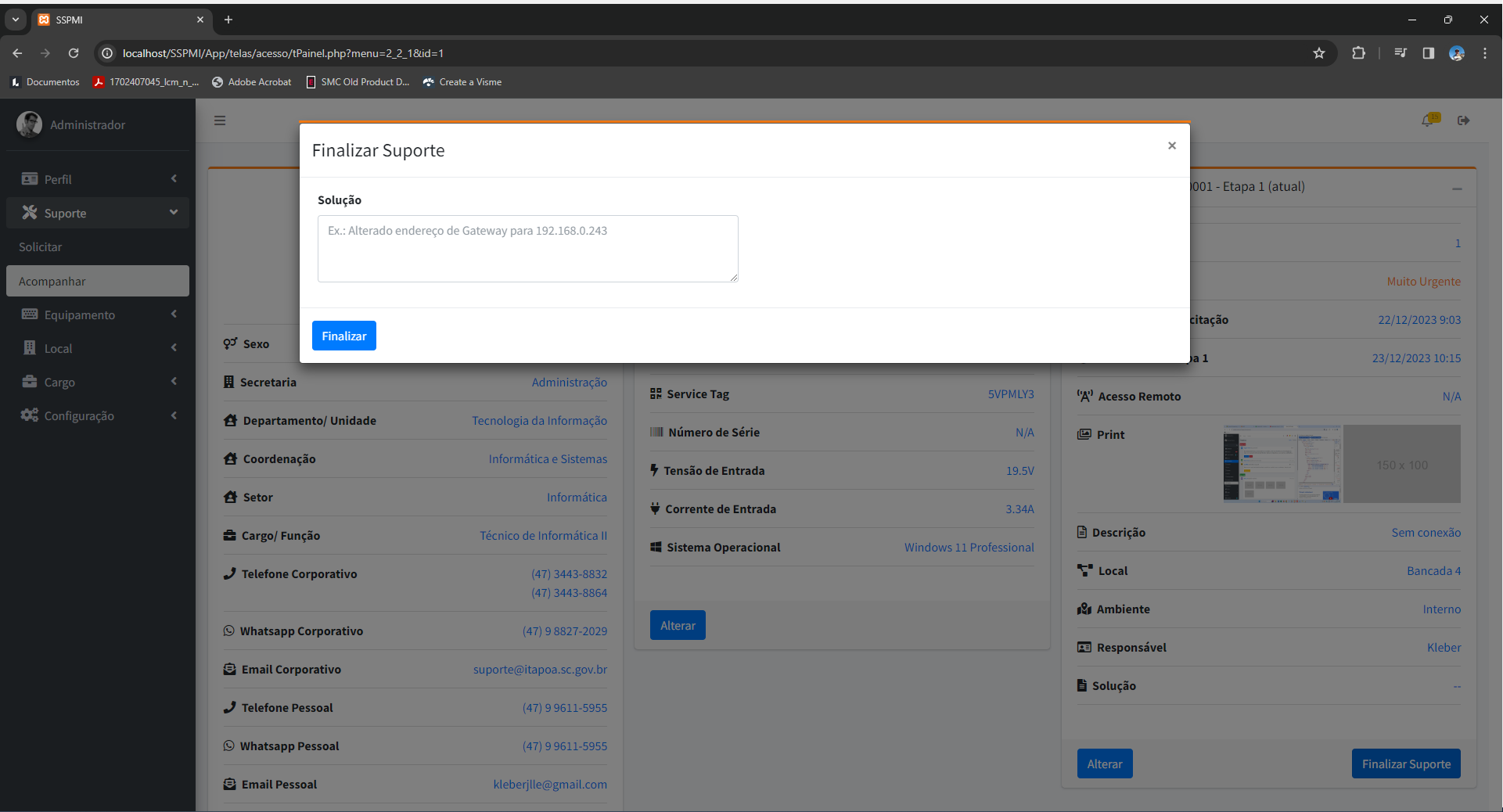


Figura - Modal "finalizarSuporte" menu2\_2\_1.php

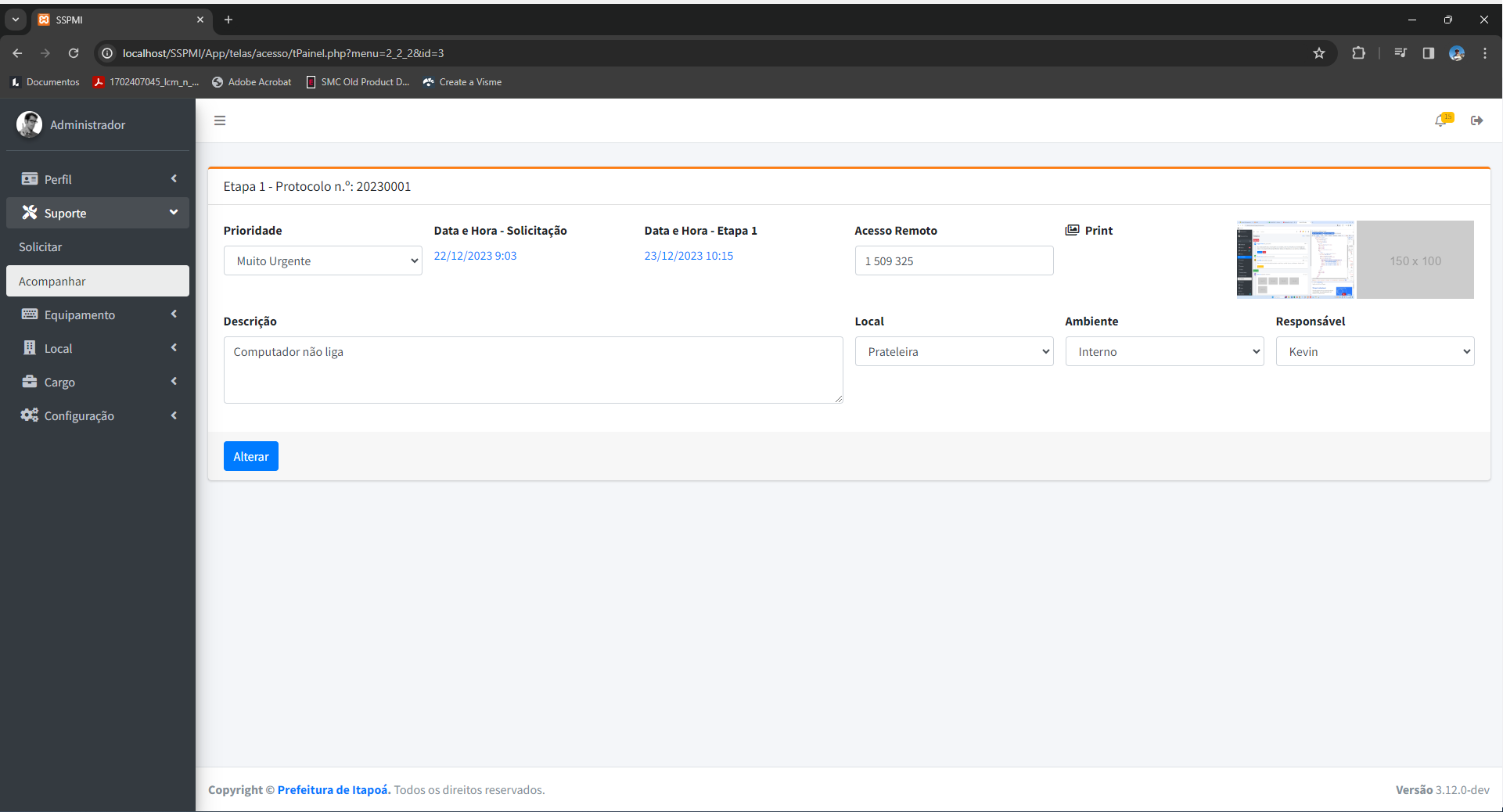


Figura - menu2\_2\_2.php

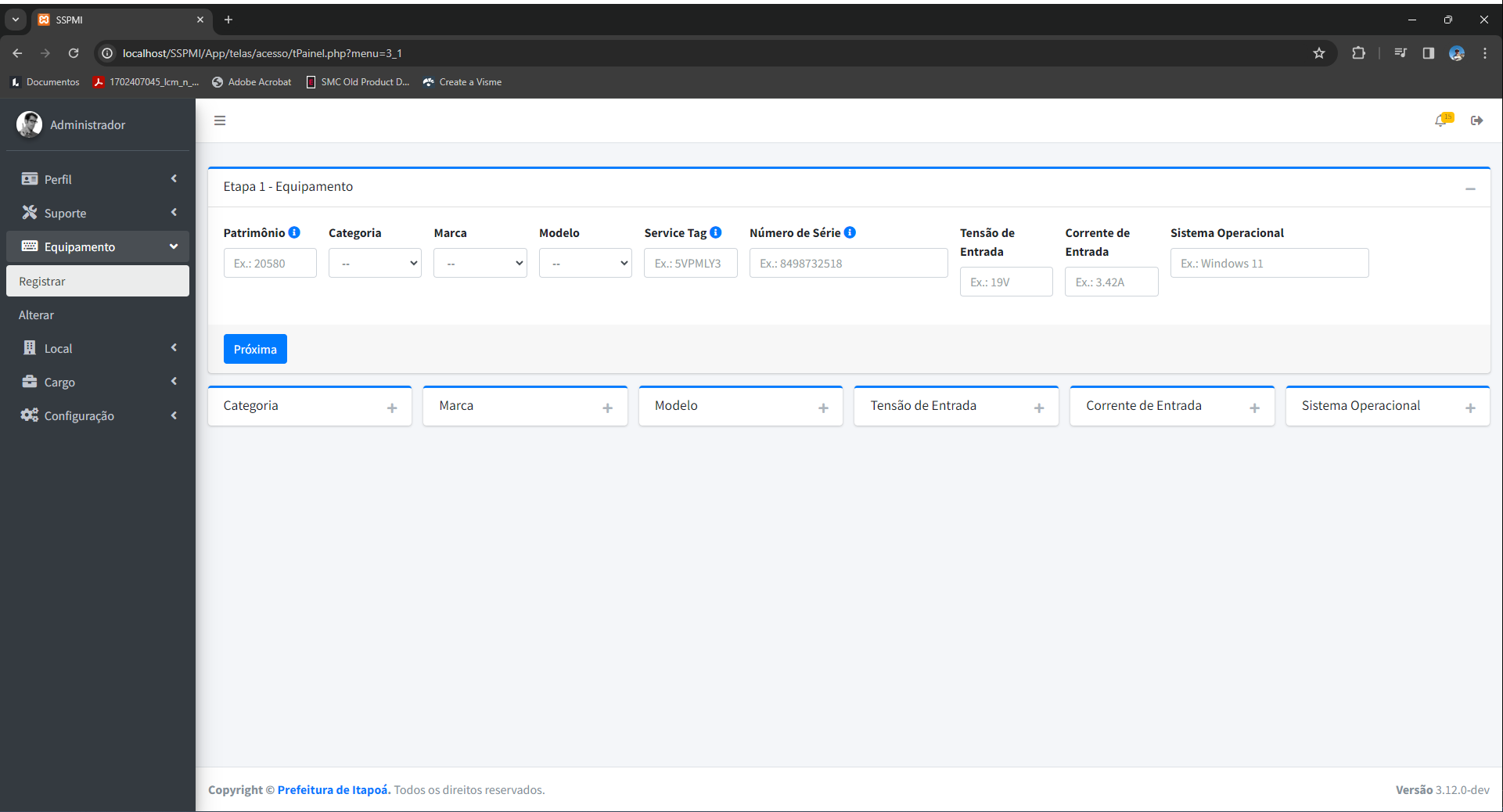


Figura - menu3\_1.php

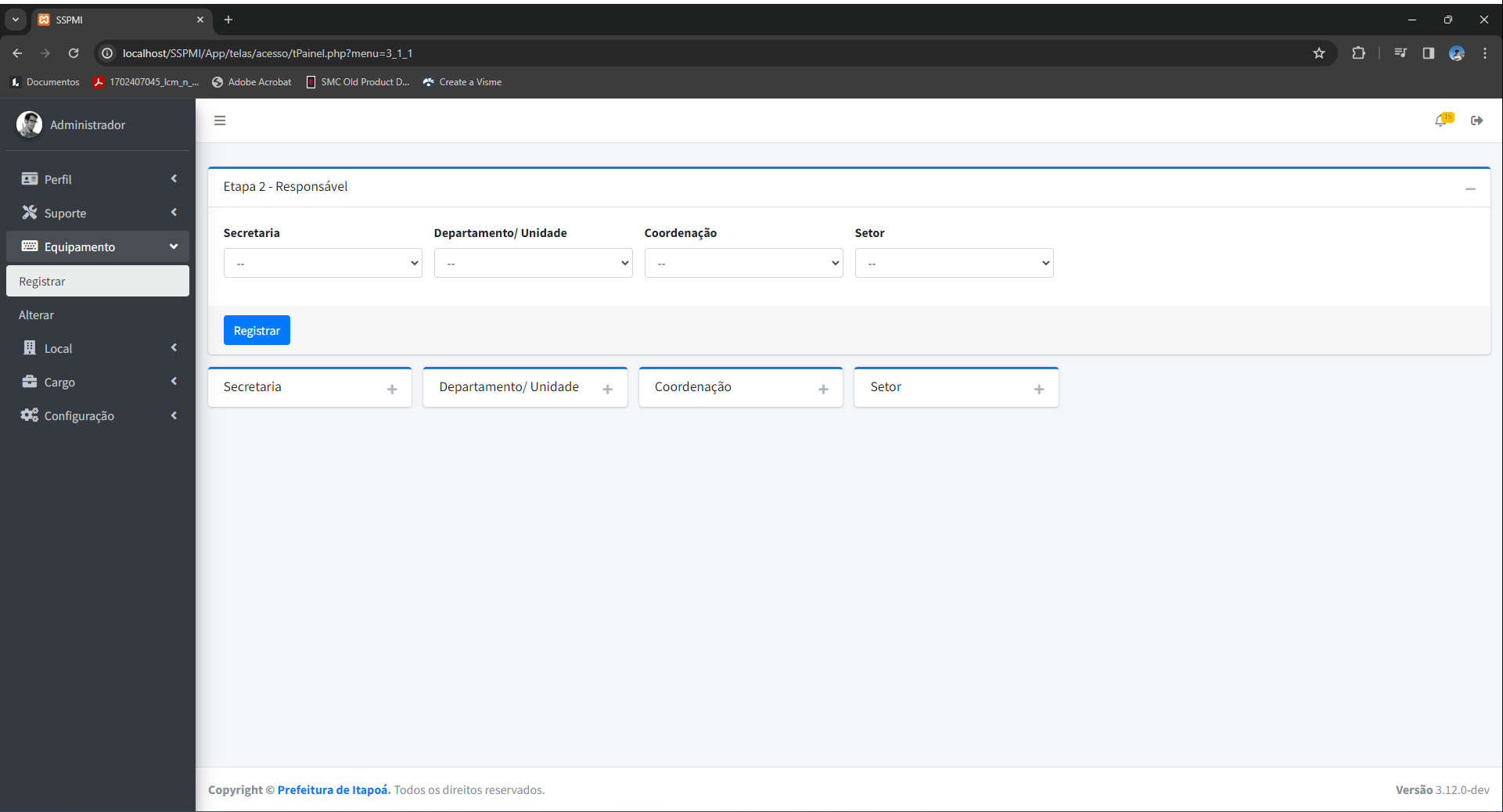


Figura - menu3\_1\_1.php

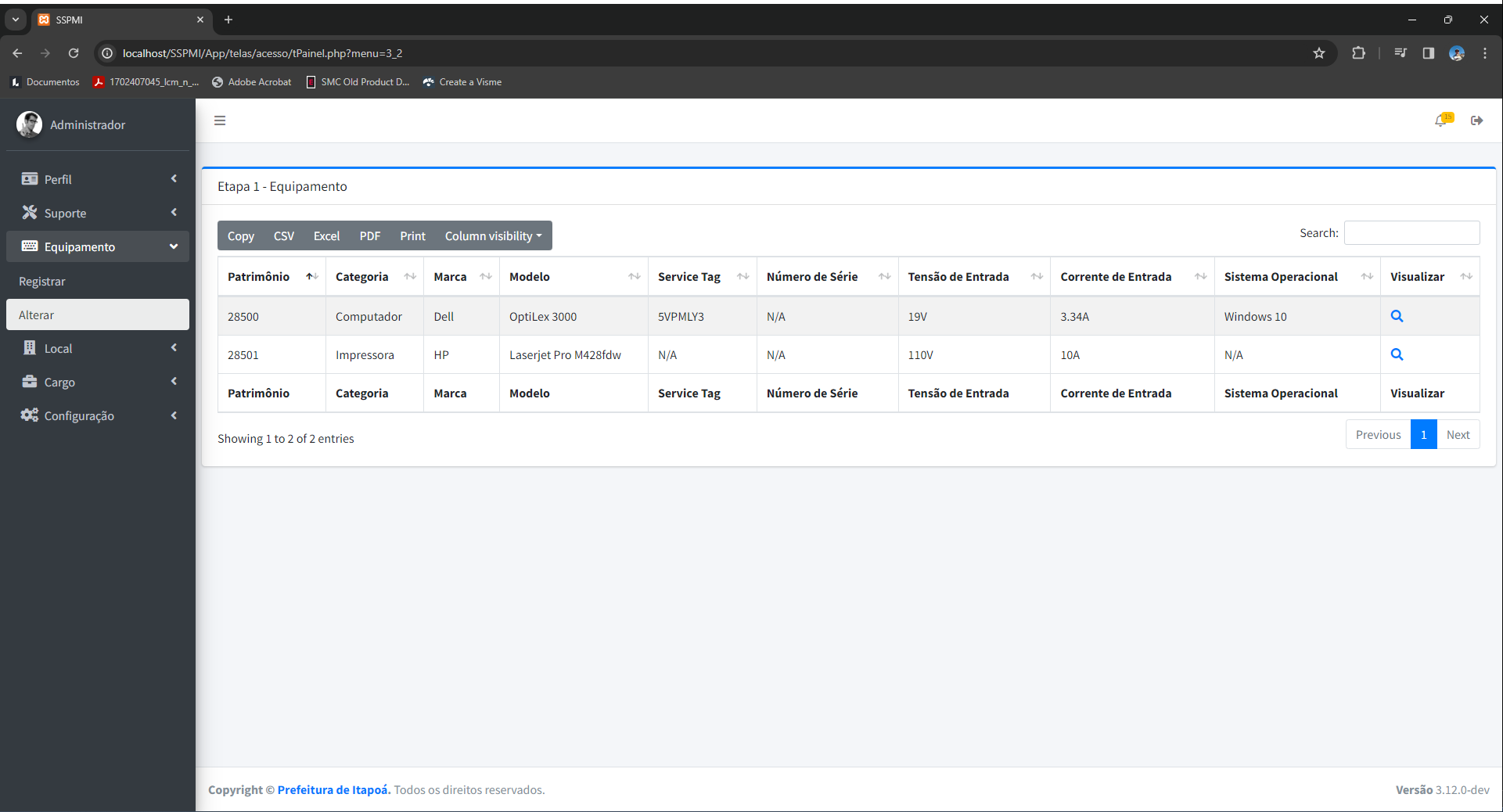


Figura - menu3\_2.php

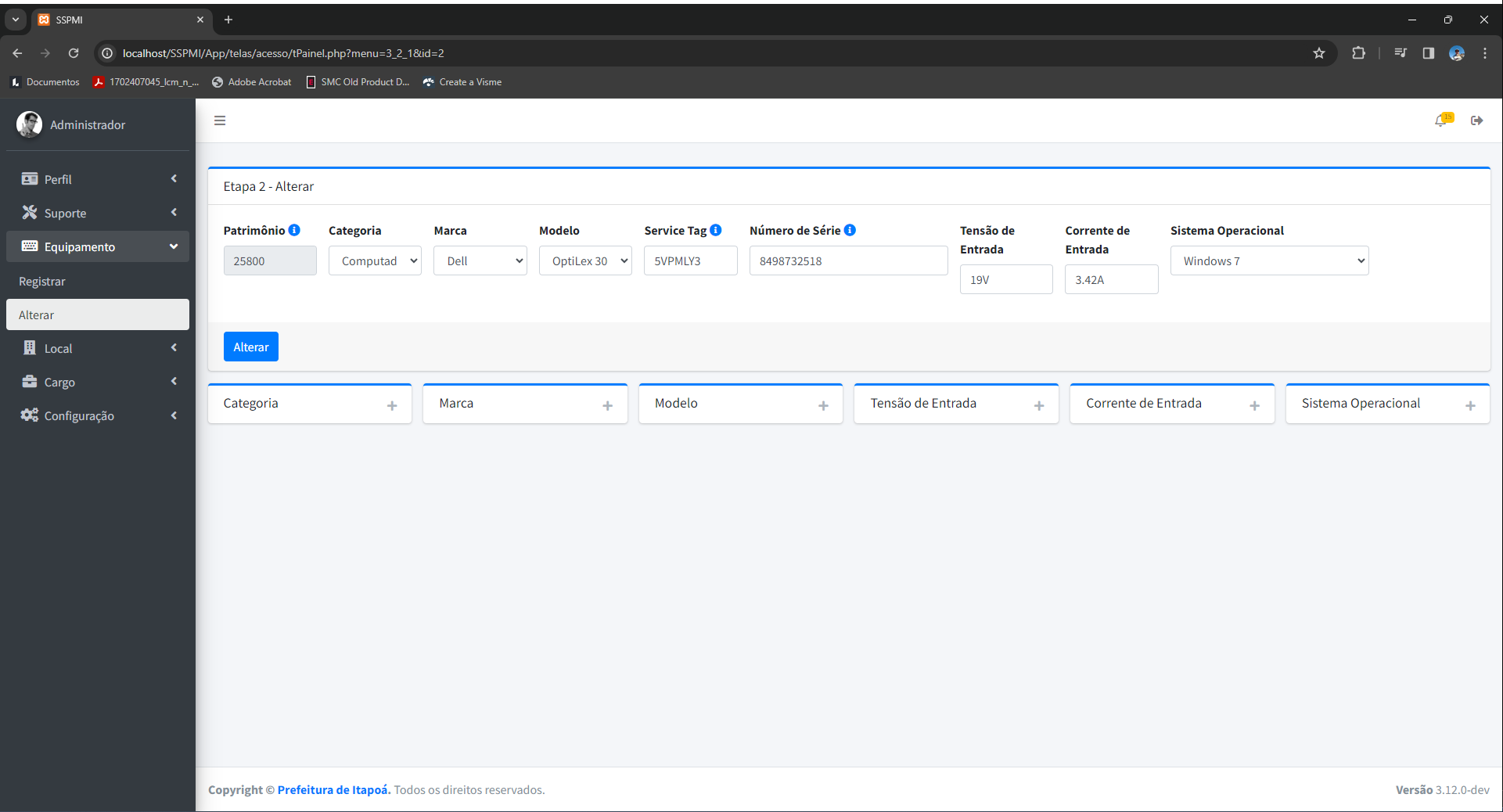


Figura - menu3\_2\_1.php

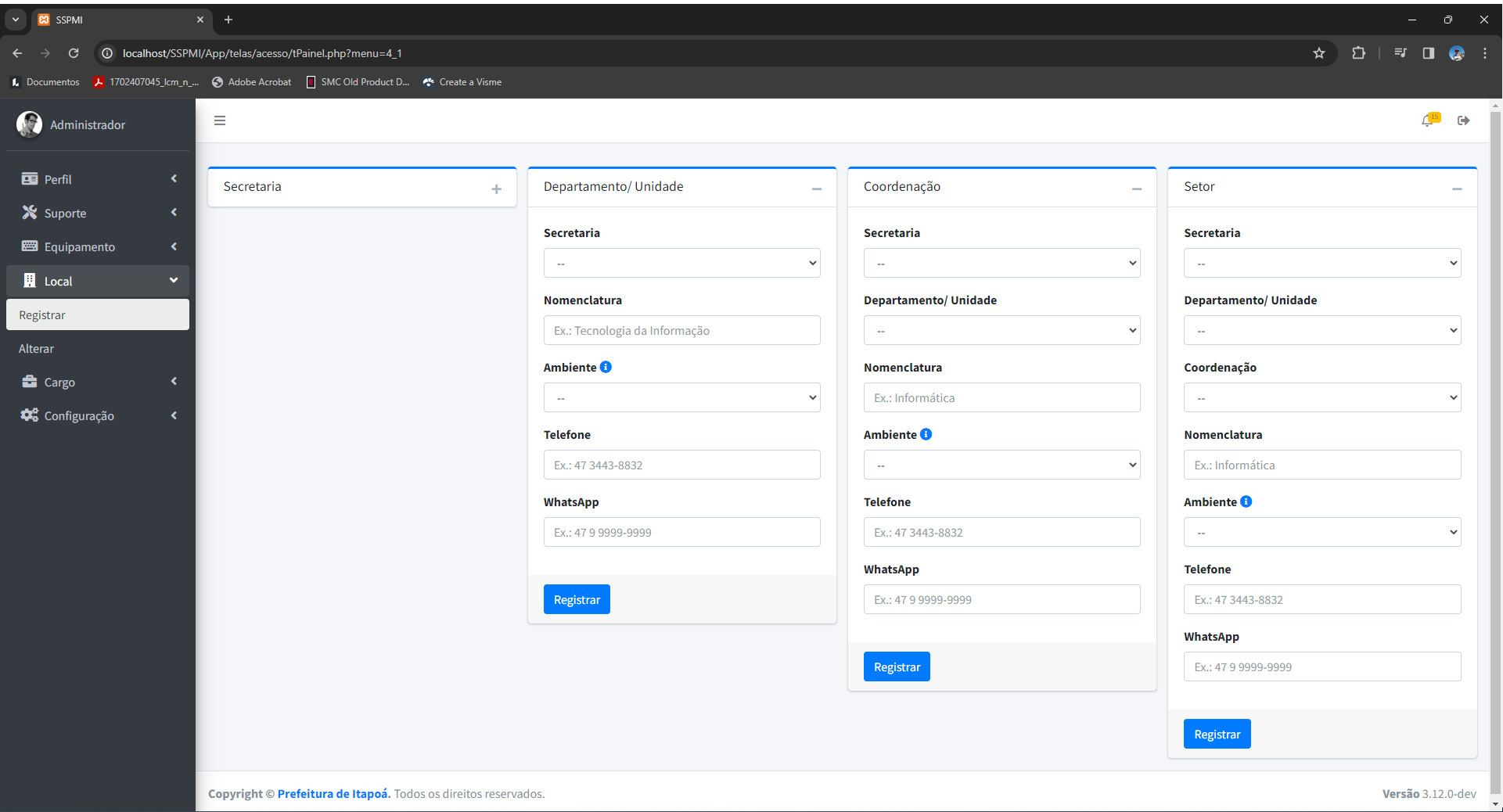


Figura - menu4\_1.php

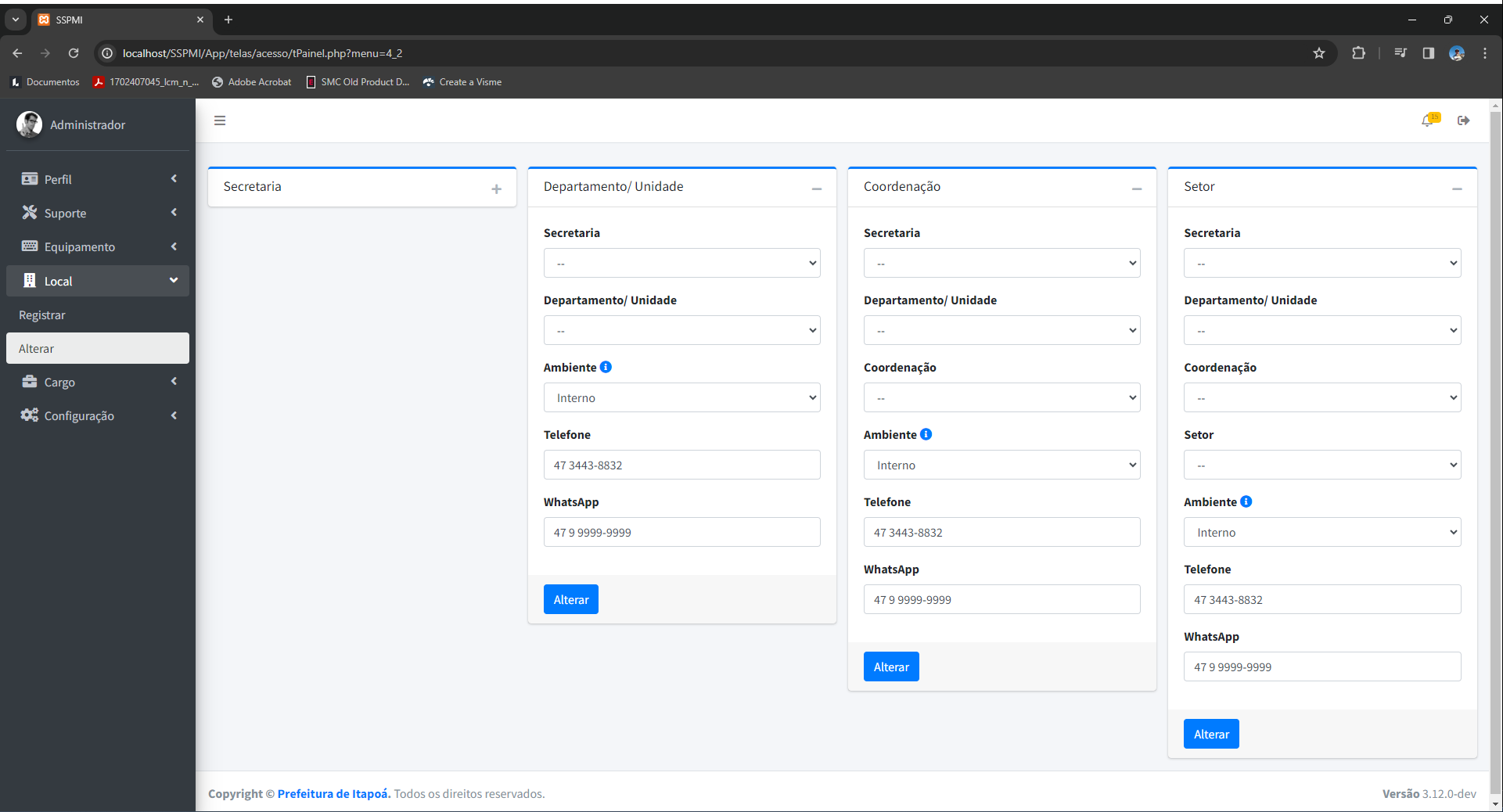


Figura - menu4\_2.php

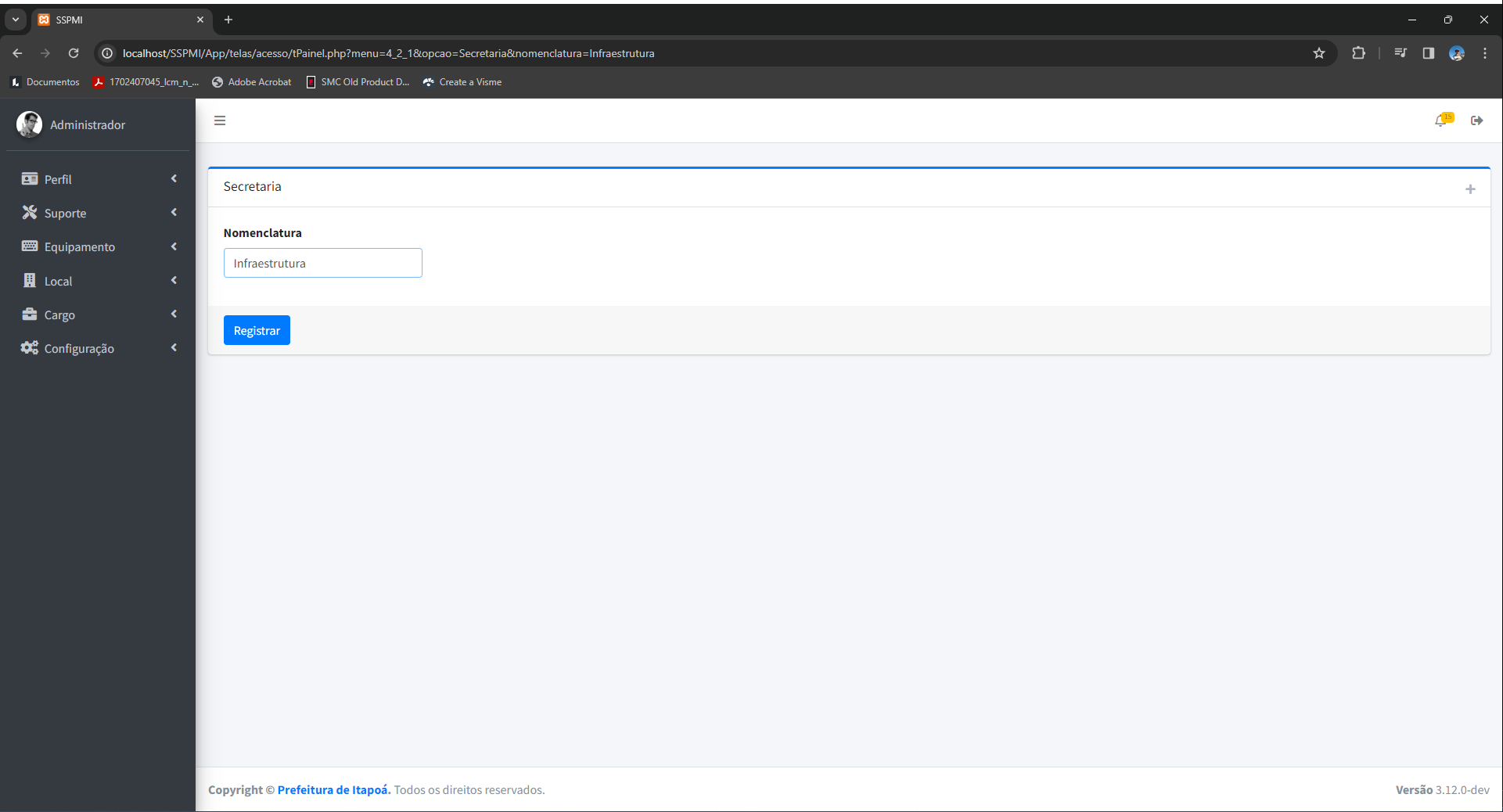


Figura - menu4\_2\_1.php

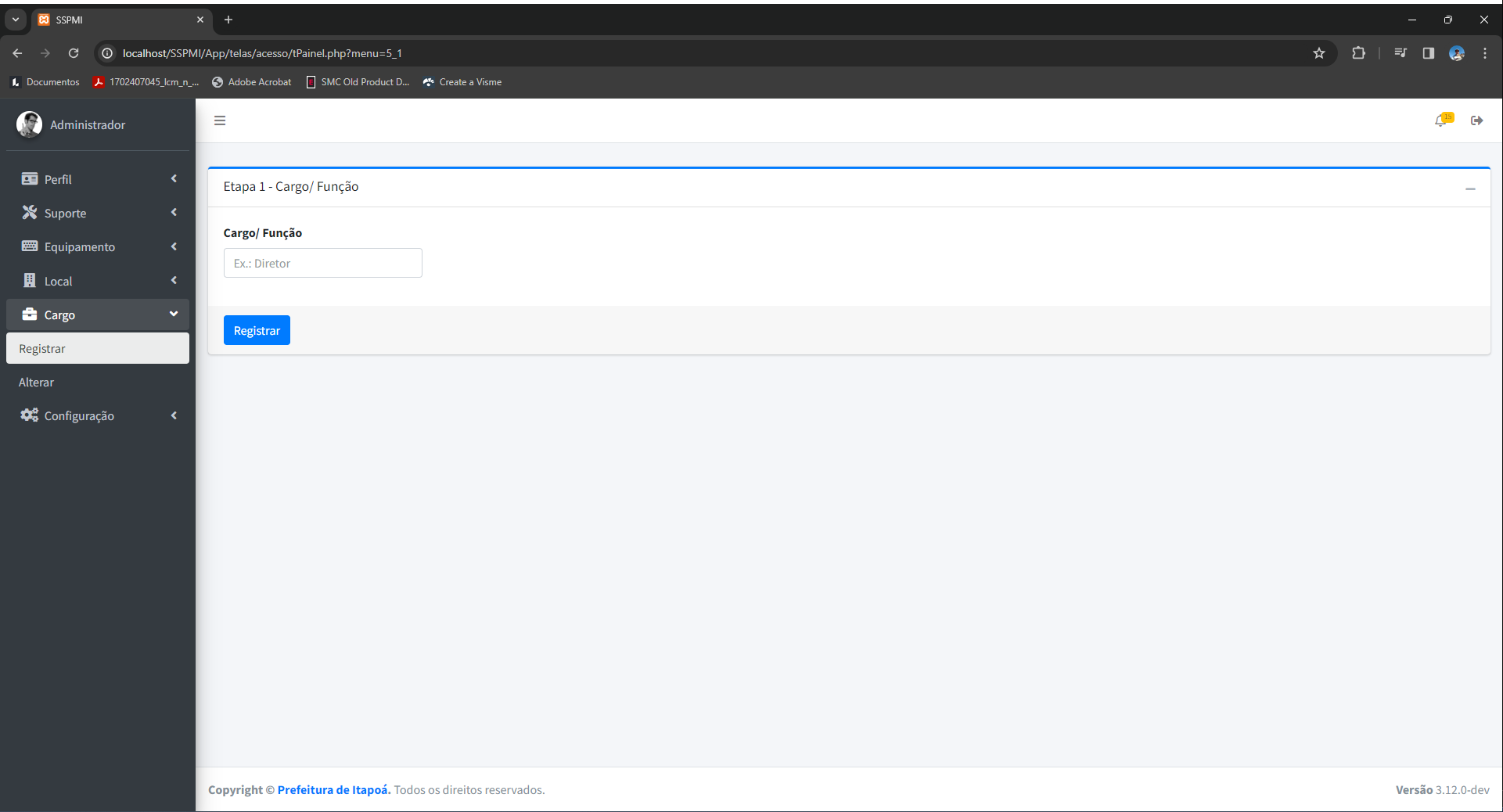


Figura - menu5\_1.php

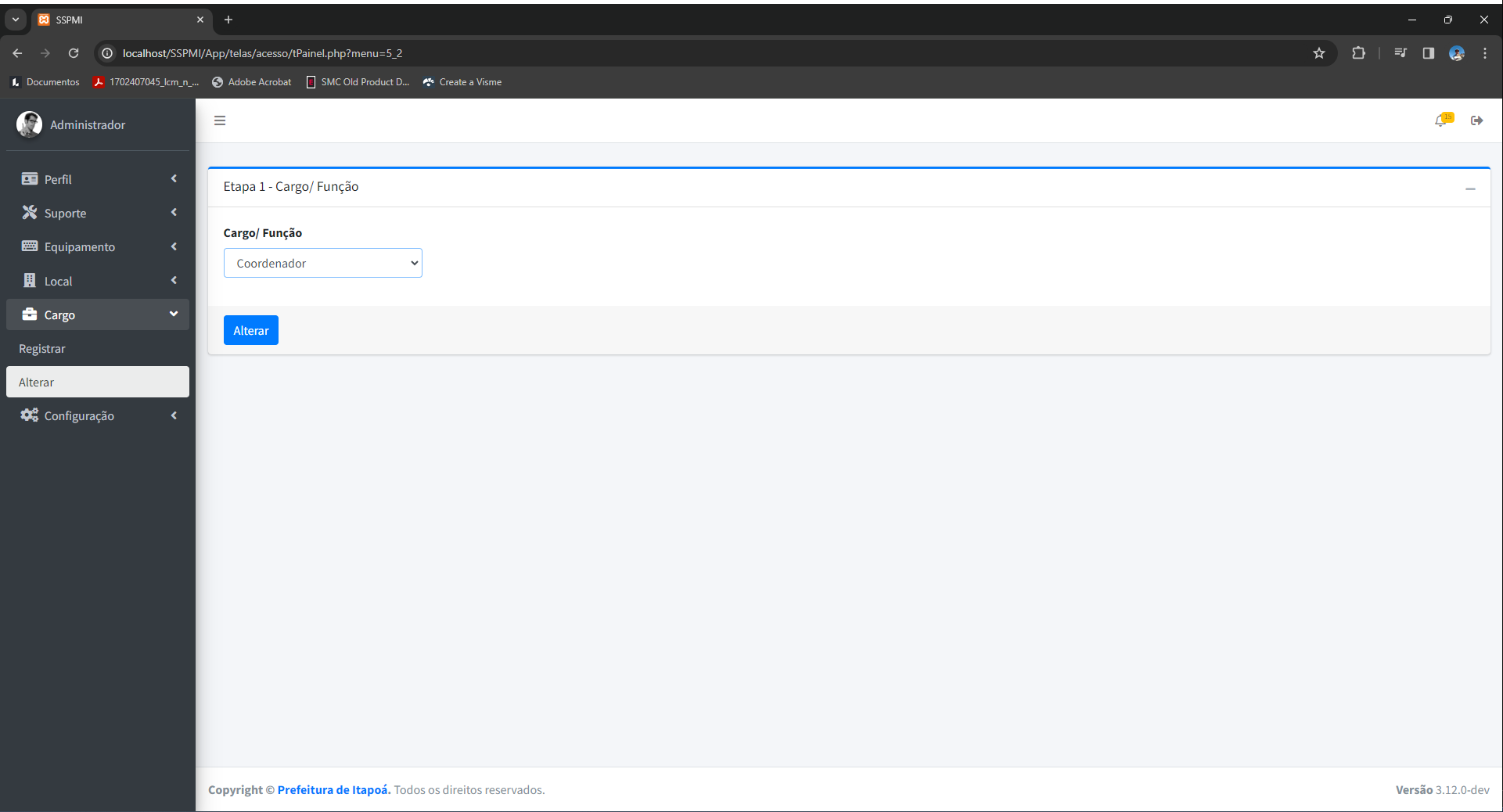


Figura - menu5\_2.php

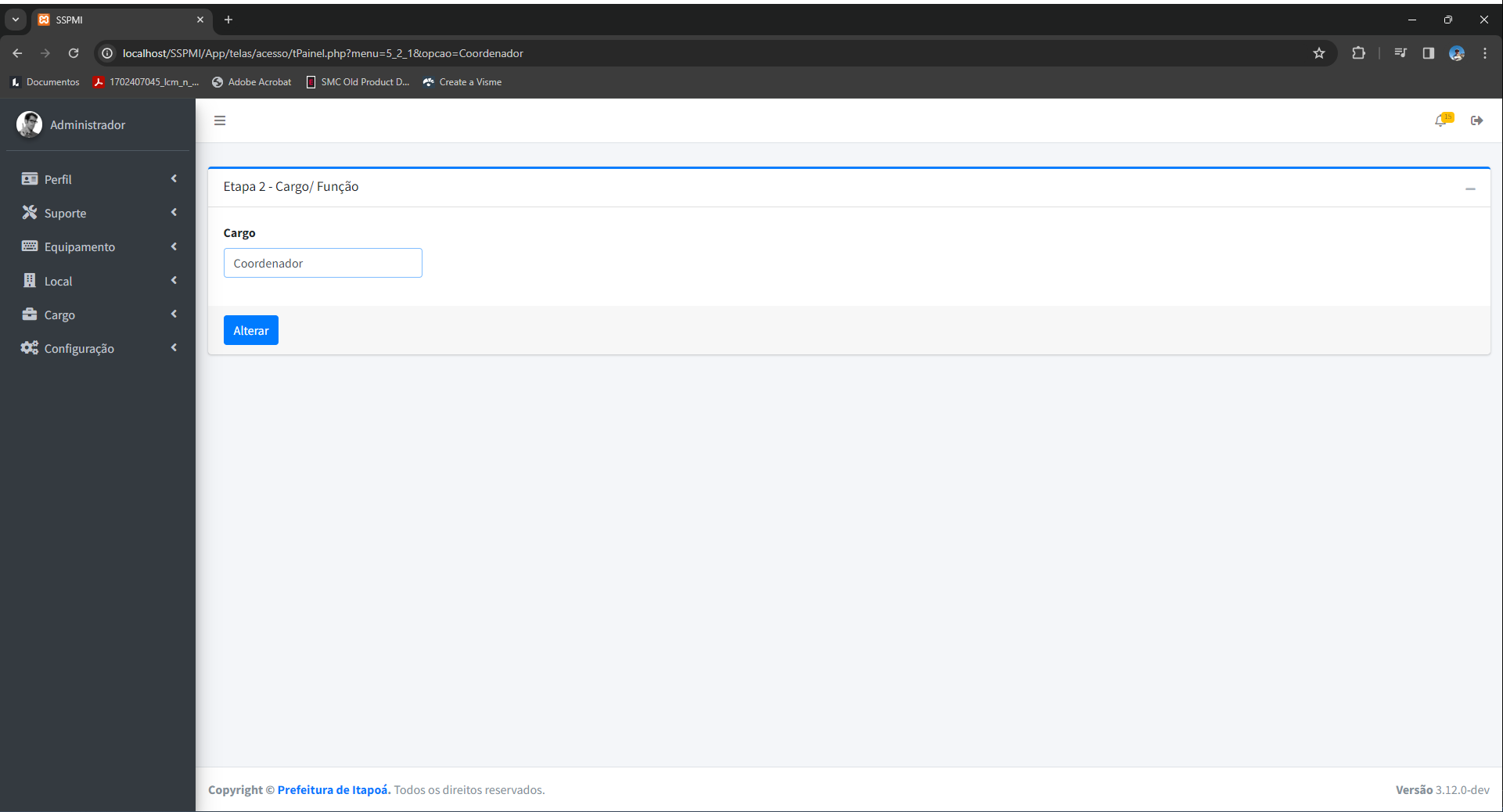


Figura - menu5\_2\_1.php

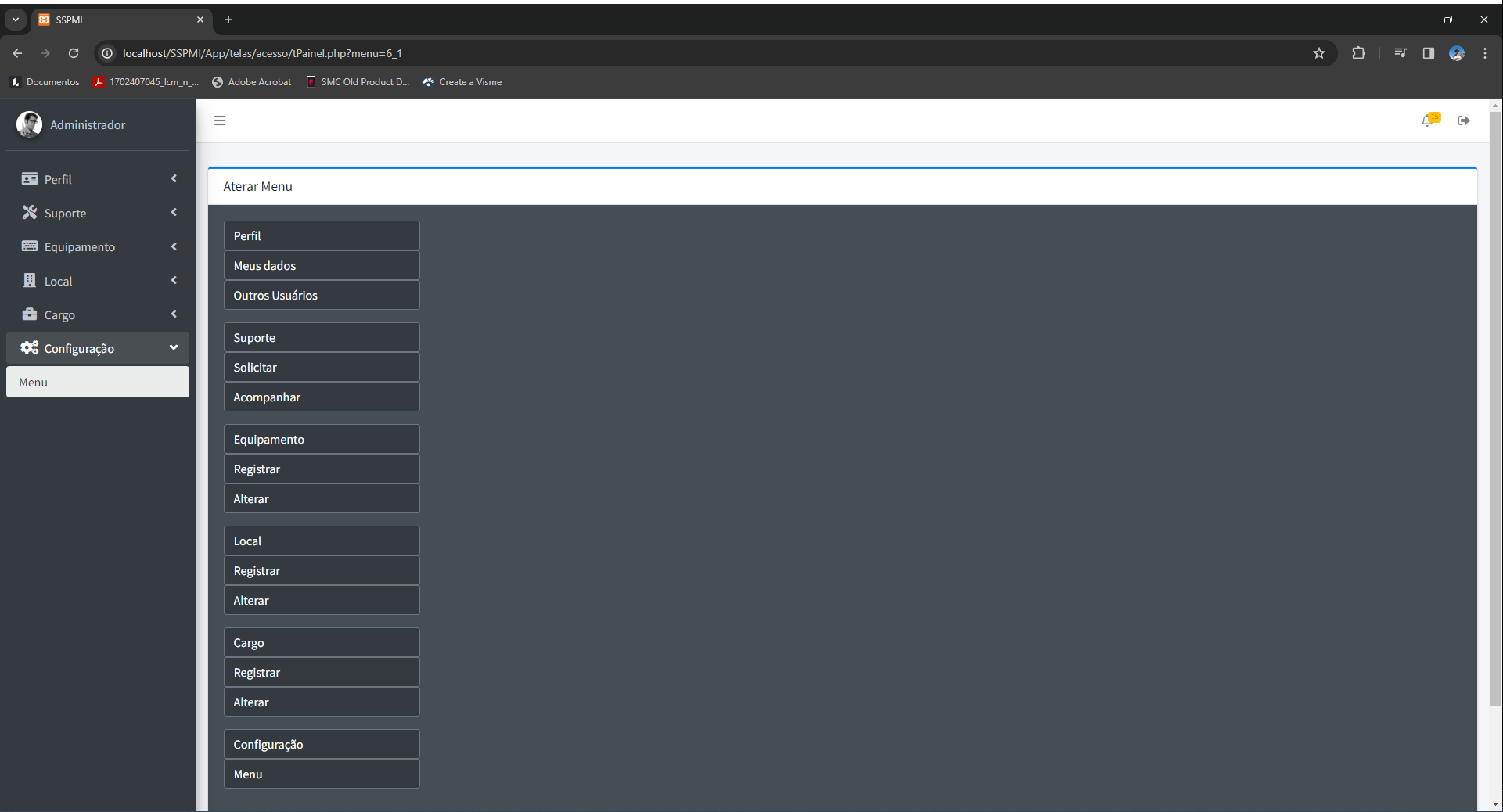


Figura - menu6\_1.php

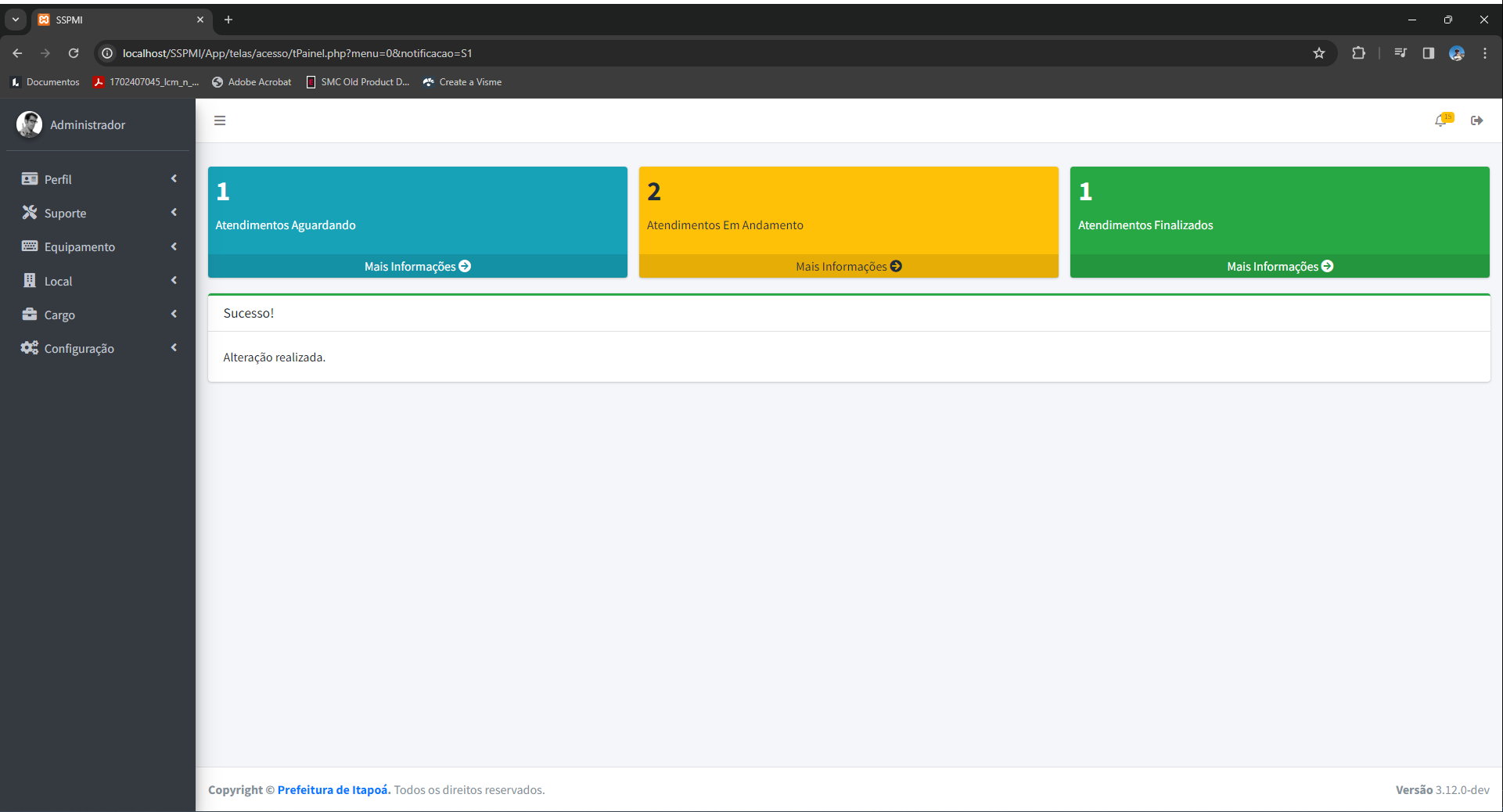


Figura - Message Box do tPainel.php

## MODELAGEM DE BD

### *SCRIPT* VERSÃO 6.1

-- MySQL Script generated by MySQL Workbench

-- Fri May 3 11:27:49 2024

-- Model: New Model Version: 1.0

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema sspmi

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema sspmi

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `sspmi` DEFAULT CHARACTER SET utf8 ;

USE `sspmi` ;

-- -----------------------------------------------------

-- Table `sspmi`.`secretaria`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`secretaria` (

`idsecretaria` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

`endereco` VARCHAR(100) NOT NULL,

PRIMARY KEY (`idsecretaria`),

UNIQUE INDEX `idsecretaria\_UNIQUE` (`idsecretaria` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`departamento`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`departamento` (

`iddepartamento` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

`endereco` VARCHAR(100) NOT NULL,

`secretaria\_idsecretaria` INT NOT NULL,

PRIMARY KEY (`iddepartamento`, `secretaria\_idsecretaria`),

UNIQUE INDEX `iddepartamento\_UNIQUE` (`iddepartamento` ASC),

INDEX `fk\_departamento\_secretaria1\_idx` (`secretaria\_idsecretaria` ASC),

CONSTRAINT `fk\_departamento\_secretaria1`

FOREIGN KEY (`secretaria\_idsecretaria`)

REFERENCES `sspmi`.`secretaria` (`idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`coordenacao`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`coordenacao` (

`idcoordenacao` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

`endereco` VARCHAR(100) NOT NULL,

`departamento\_secretaria\_idsecretaria` INT NOT NULL,

PRIMARY KEY (`idcoordenacao`, `departamento\_secretaria\_idsecretaria`),

UNIQUE INDEX `idcoordenacao\_UNIQUE` (`idcoordenacao` ASC),

INDEX `fk\_coordenacao\_departamento1\_idx` (`departamento\_secretaria\_idsecretaria` ASC),

CONSTRAINT `fk\_coordenacao\_departamento1`

FOREIGN KEY (`departamento\_secretaria\_idsecretaria`)

REFERENCES `sspmi`.`departamento` (`secretaria\_idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`setor`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`setor` (

`idsetor` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

`endereco` VARCHAR(100) NOT NULL,

`coordenacao\_departamento\_iddepartamento` INT NULL,

`coordenacao\_departamento\_secretaria\_idsecretaria` INT NULL,

PRIMARY KEY (`idsetor`),

UNIQUE INDEX `idsetor\_UNIQUE` (`idsetor` ASC),

INDEX `fk\_setor\_coordenacao1\_idx` (`coordenacao\_departamento\_iddepartamento` ASC, `coordenacao\_departamento\_secretaria\_idsecretaria` ASC),

CONSTRAINT `fk\_setor\_coordenacao1`

FOREIGN KEY (`coordenacao\_departamento\_secretaria\_idsecretaria`)

REFERENCES `sspmi`.`coordenacao` (`departamento\_secretaria\_idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`telefone`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`telefone` (

`idtelefone` INT NOT NULL AUTO\_INCREMENT,

`whatsApp` TINYINT NOT NULL DEFAULT 0,

`numero` VARCHAR(11) NULL,

PRIMARY KEY (`idtelefone`),

UNIQUE INDEX `idtelefone\_UNIQUE` (`idtelefone` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`cargo`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`cargo` (

`idcargo` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

PRIMARY KEY (`idcargo`),

UNIQUE INDEX `idcargo\_UNIQUE` (`idcargo` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`email`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`email` (

`idemail` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NULL,

PRIMARY KEY (`idemail`),

UNIQUE INDEX `idemail\_UNIQUE` (`idemail` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`permissao`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`permissao` (

`idpermissao` INT NOT NULL AUTO\_INCREMENT,

`nivel` INT NOT NULL,

`nomenclatura` VARCHAR(100) NOT NULL,

PRIMARY KEY (`idpermissao`),

UNIQUE INDEX `idpermissao\_UNIQUE` (`idpermissao` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC),

UNIQUE INDEX `nivel\_UNIQUE` (`nivel` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`usuario`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`usuario` (

`idusuario` INT NOT NULL AUTO\_INCREMENT,

`nome` VARCHAR(100) NOT NULL,

`sobrenome` VARCHAR(200) NOT NULL,

`sexo` CHAR(1) NOT NULL,

`imagem` VARCHAR(45) NOT NULL DEFAULT 'default.jpg',

`situacao` CHAR(1) NOT NULL DEFAULT 0,

`setor\_idsetor` INT NULL,

`coordenacao\_idcoordenacao` INT NULL,

`departamento\_iddepartamento` INT NULL,

`secretaria\_idsecretaria` INT NOT NULL,

`telefone\_idtelefone` INT NULL,

`cargo\_idcargo` INT NOT NULL,

`email\_idemail` INT NOT NULL,

`permissao\_idpermissao` INT NOT NULL,

PRIMARY KEY (`idusuario`, `secretaria\_idsecretaria`, `cargo\_idcargo`, `email\_idemail`, `permissao\_idpermissao`),

UNIQUE INDEX `idusuario\_UNIQUE` (`idusuario` ASC),

INDEX `fk\_usuario\_setor1\_idx` (`setor\_idsetor` ASC),

INDEX `fk\_usuario\_coordenacao1\_idx` (`coordenacao\_idcoordenacao` ASC),

INDEX `fk\_usuario\_departamento1\_idx` (`departamento\_iddepartamento` ASC),

INDEX `fk\_usuario\_secretaria1\_idx` (`secretaria\_idsecretaria` ASC),

INDEX `fk\_usuario\_telefone1\_idx` (`telefone\_idtelefone` ASC),

INDEX `fk\_usuario\_cargo1\_idx` (`cargo\_idcargo` ASC),

INDEX `fk\_usuario\_email1\_idx` (`email\_idemail` ASC),

INDEX `fk\_usuario\_permissao1\_idx` (`permissao\_idpermissao` ASC),

CONSTRAINT `fk\_usuario\_setor1`

FOREIGN KEY (`setor\_idsetor`)

REFERENCES `sspmi`.`setor` (`idsetor`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_coordenacao1`

FOREIGN KEY (`coordenacao\_idcoordenacao`)

REFERENCES `sspmi`.`coordenacao` (`idcoordenacao`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_departamento1`

FOREIGN KEY (`departamento\_iddepartamento`)

REFERENCES `sspmi`.`departamento` (`iddepartamento`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_secretaria1`

FOREIGN KEY (`secretaria\_idsecretaria`)

REFERENCES `sspmi`.`secretaria` (`idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_telefone1`

FOREIGN KEY (`telefone\_idtelefone`)

REFERENCES `sspmi`.`telefone` (`idtelefone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_cargo1`

FOREIGN KEY (`cargo\_idcargo`)

REFERENCES `sspmi`.`cargo` (`idcargo`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_email1`

FOREIGN KEY (`email\_idemail`)

REFERENCES `sspmi`.`email` (`idemail`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_usuario\_permissao1`

FOREIGN KEY (`permissao\_idpermissao`)

REFERENCES `sspmi`.`permissao` (`idpermissao`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`historico`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`historico` (

`idhistorico` INT NOT NULL AUTO\_INCREMENT,

`pagina` VARCHAR(45) NOT NULL,

`acao` VARCHAR(45) NOT NULL,

`campo` VARCHAR(100) NOT NULL,

`valorAtual` VARCHAR(100) NOT NULL,

`valorAnterior` VARCHAR(100) NOT NULL,

`dataHora` TIMESTAMP NOT NULL,

`ip` VARCHAR(12) NOT NULL,

`navegador` VARCHAR(100) NOT NULL,

`usuario\_idusuario` INT NOT NULL,

`usuario\_secretaria\_idsecretaria` INT NOT NULL,

PRIMARY KEY (`idhistorico`, `usuario\_idusuario`, `usuario\_secretaria\_idsecretaria`),

UNIQUE INDEX `idhistorico\_UNIQUE` (`idhistorico` ASC),

INDEX `fk\_historico\_usuario1\_idx` (`usuario\_idusuario` ASC, `usuario\_secretaria\_idsecretaria` ASC),

CONSTRAINT `fk\_historico\_usuario1`

FOREIGN KEY (`usuario\_idusuario` , `usuario\_secretaria\_idsecretaria`)

REFERENCES `sspmi`.`usuario` (`idusuario` , `secretaria\_idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`telefone\_has\_secretaria`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`telefone\_has\_secretaria` (

`telefone\_idtelefone` INT NOT NULL,

`secretaria\_idsecretaria` INT NOT NULL,

PRIMARY KEY (`telefone\_idtelefone`, `secretaria\_idsecretaria`),

INDEX `fk\_telefone\_has\_secretaria\_secretaria1\_idx` (`secretaria\_idsecretaria` ASC),

INDEX `fk\_telefone\_has\_secretaria\_telefone1\_idx` (`telefone\_idtelefone` ASC),

CONSTRAINT `fk\_telefone\_has\_secretaria\_telefone1`

FOREIGN KEY (`telefone\_idtelefone`)

REFERENCES `sspmi`.`telefone` (`idtelefone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_telefone\_has\_secretaria\_secretaria1`

FOREIGN KEY (`secretaria\_idsecretaria`)

REFERENCES `sspmi`.`secretaria` (`idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`telefone\_has\_departamento`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`telefone\_has\_departamento` (

`telefone\_idtelefone` INT NOT NULL,

`departamento\_iddepartamento` INT NOT NULL,

PRIMARY KEY (`telefone\_idtelefone`, `departamento\_iddepartamento`),

INDEX `fk\_telefone\_has\_departamento\_departamento1\_idx` (`departamento\_iddepartamento` ASC),

INDEX `fk\_telefone\_has\_departamento\_telefone1\_idx` (`telefone\_idtelefone` ASC),

CONSTRAINT `fk\_telefone\_has\_departamento\_telefone1`

FOREIGN KEY (`telefone\_idtelefone`)

REFERENCES `sspmi`.`telefone` (`idtelefone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_telefone\_has\_departamento\_departamento1`

FOREIGN KEY (`departamento\_iddepartamento`)

REFERENCES `sspmi`.`departamento` (`iddepartamento`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`telefone\_has\_coordenacao`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`telefone\_has\_coordenacao` (

`telefone\_idtelefone` INT NOT NULL,

`coordenacao\_idcoordenacao` INT NOT NULL,

PRIMARY KEY (`telefone\_idtelefone`, `coordenacao\_idcoordenacao`),

INDEX `fk\_telefone\_has\_coordenacao\_coordenacao1\_idx` (`coordenacao\_idcoordenacao` ASC),

INDEX `fk\_telefone\_has\_coordenacao\_telefone1\_idx` (`telefone\_idtelefone` ASC),

CONSTRAINT `fk\_telefone\_has\_coordenacao\_telefone1`

FOREIGN KEY (`telefone\_idtelefone`)

REFERENCES `sspmi`.`telefone` (`idtelefone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_telefone\_has\_coordenacao\_coordenacao1`

FOREIGN KEY (`coordenacao\_idcoordenacao`)

REFERENCES `sspmi`.`coordenacao` (`idcoordenacao`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`telefone\_has\_setor`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`telefone\_has\_setor` (

`telefone\_idtelefone` INT NOT NULL,

`setor\_idsetor` INT NOT NULL,

PRIMARY KEY (`telefone\_idtelefone`, `setor\_idsetor`),

INDEX `fk\_telefone\_has\_setor\_setor1\_idx` (`setor\_idsetor` ASC),

INDEX `fk\_telefone\_has\_setor\_telefone1\_idx` (`telefone\_idtelefone` ASC),

CONSTRAINT `fk\_telefone\_has\_setor\_telefone1`

FOREIGN KEY (`telefone\_idtelefone`)

REFERENCES `sspmi`.`telefone` (`idtelefone`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_telefone\_has\_setor\_setor1`

FOREIGN KEY (`setor\_idsetor`)

REFERENCES `sspmi`.`setor` (`idsetor`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`email\_has\_setor`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`email\_has\_setor` (

`email\_idemail` INT NOT NULL,

`setor\_idsetor` INT NOT NULL,

PRIMARY KEY (`email\_idemail`, `setor\_idsetor`),

INDEX `fk\_email\_has\_setor\_setor1\_idx` (`setor\_idsetor` ASC),

INDEX `fk\_email\_has\_setor\_email1\_idx` (`email\_idemail` ASC),

CONSTRAINT `fk\_email\_has\_setor\_email1`

FOREIGN KEY (`email\_idemail`)

REFERENCES `sspmi`.`email` (`idemail`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_email\_has\_setor\_setor1`

FOREIGN KEY (`setor\_idsetor`)

REFERENCES `sspmi`.`setor` (`idsetor`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`email\_has\_coordenacao`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`email\_has\_coordenacao` (

`email\_idemail` INT NOT NULL,

`coordenacao\_idcoordenacao` INT NOT NULL,

PRIMARY KEY (`email\_idemail`, `coordenacao\_idcoordenacao`),

INDEX `fk\_email\_has\_coordenacao\_coordenacao1\_idx` (`coordenacao\_idcoordenacao` ASC),

INDEX `fk\_email\_has\_coordenacao\_email1\_idx` (`email\_idemail` ASC),

CONSTRAINT `fk\_email\_has\_coordenacao\_email1`

FOREIGN KEY (`email\_idemail`)

REFERENCES `sspmi`.`email` (`idemail`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_email\_has\_coordenacao\_coordenacao1`

FOREIGN KEY (`coordenacao\_idcoordenacao`)

REFERENCES `sspmi`.`coordenacao` (`idcoordenacao`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`email\_has\_departamento`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`email\_has\_departamento` (

`email\_idemail` INT NOT NULL,

`departamento\_iddepartamento` INT NOT NULL,

PRIMARY KEY (`email\_idemail`, `departamento\_iddepartamento`),

INDEX `fk\_email\_has\_departamento\_departamento1\_idx` (`departamento\_iddepartamento` ASC),

INDEX `fk\_email\_has\_departamento\_email1\_idx` (`email\_idemail` ASC),

CONSTRAINT `fk\_email\_has\_departamento\_email1`

FOREIGN KEY (`email\_idemail`)

REFERENCES `sspmi`.`email` (`idemail`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_email\_has\_departamento\_departamento1`

FOREIGN KEY (`departamento\_iddepartamento`)

REFERENCES `sspmi`.`departamento` (`iddepartamento`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`email\_has\_secretaria`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`email\_has\_secretaria` (

`email\_idemail` INT NOT NULL,

`secretaria\_idsecretaria` INT NOT NULL,

PRIMARY KEY (`email\_idemail`, `secretaria\_idsecretaria`),

INDEX `fk\_email\_has\_secretaria\_secretaria1\_idx` (`secretaria\_idsecretaria` ASC),

INDEX `fk\_email\_has\_secretaria\_email1\_idx` (`email\_idemail` ASC),

CONSTRAINT `fk\_email\_has\_secretaria\_email1`

FOREIGN KEY (`email\_idemail`)

REFERENCES `sspmi`.`email` (`idemail`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_email\_has\_secretaria\_secretaria1`

FOREIGN KEY (`secretaria\_idsecretaria`)

REFERENCES `sspmi`.`secretaria` (`idsecretaria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`prioridade`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`prioridade` (

`idprioridade` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(45) NOT NULL DEFAULT 'normal',

PRIMARY KEY (`idprioridade`),

UNIQUE INDEX `idprioridade\_UNIQUE` (`idprioridade` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`protocolo`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`protocolo` (

`idprotocolo` INT NOT NULL AUTO\_INCREMENT,

`ano` YEAR(4) NOT NULL,

`dataHoraAbertura` TIMESTAMP NOT NULL,

`dataHoraEncerramento` DATETIME NULL,

`nomeDoRepresentado` VARCHAR(45) NULL,

`sobrenomeDoRepresentado` VARCHAR(100) NULL,

`telefoneDoRepresentado` VARCHAR(45) NULL,

`whatsAppDoRepresentado` VARCHAR(11) NULL,

`emailDoRepresentado` VARCHAR(100) NULL,

`usuario\_idusuario` INT NOT NULL,

`prioridade\_idprioridade` INT NOT NULL,

PRIMARY KEY (`idprotocolo`, `usuario\_idusuario`, `prioridade\_idprioridade`),

UNIQUE INDEX `idprotocolo\_UNIQUE` (`idprotocolo` ASC),

INDEX `fk\_protocolo\_usuario1\_idx` (`usuario\_idusuario` ASC),

INDEX `fk\_protocolo\_prioridade1\_idx` (`prioridade\_idprioridade` ASC),

CONSTRAINT `fk\_protocolo\_usuario1`

FOREIGN KEY (`usuario\_idusuario`)

REFERENCES `sspmi`.`usuario` (`idusuario`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_protocolo\_prioridade1`

FOREIGN KEY (`prioridade\_idprioridade`)

REFERENCES `sspmi`.`prioridade` (`idprioridade`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`categoria`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`categoria` (

`idcategoria` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL DEFAULT 'indefinida',

PRIMARY KEY (`idcategoria`),

UNIQUE INDEX `idcategoria\_UNIQUE` (`idcategoria` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`sistemaOperacional`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`sistemaOperacional` (

`idsistemaOperacional` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

PRIMARY KEY (`idsistemaOperacional`),

UNIQUE INDEX `idsistemmaOperacional\_UNIQUE` (`idsistemaOperacional` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`tensao`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`tensao` (

`idtensao` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(10) NOT NULL,

PRIMARY KEY (`idtensao`),

UNIQUE INDEX `idtensao\_UNIQUE` (`idtensao` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`corrente`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`corrente` (

`idcorrente` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(10) NOT NULL,

PRIMARY KEY (`idcorrente`),

UNIQUE INDEX `idcorrente\_UNIQUE` (`idcorrente` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`ambiente`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`ambiente` (

`idambiente` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(45) NOT NULL,

PRIMARY KEY (`idambiente`),

UNIQUE INDEX `idambiente\_UNIQUE` (`idambiente` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`local`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`local` (

`idlocal` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(45) NOT NULL,

PRIMARY KEY (`idlocal`),

UNIQUE INDEX `idlocal\_UNIQUE` (`idlocal` ASC),

UNIQUE INDEX `nomenclatura\_UNIQUE` (`nomenclatura` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`equipamento`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`equipamento` (

`idequipamento` INT NOT NULL AUTO\_INCREMENT,

`patrimonio` VARCHAR(10) NOT NULL,

`categoria\_idcategoria` INT NOT NULL,

`sistemaOperacional\_idsistemmaOperacional` INT NULL,

`tensao\_idtensao` INT NULL,

`corrente\_idcorrente` INT NULL,

`ambiente\_idambiente` INT NULL,

`local\_idlocal` INT NOT NULL,

PRIMARY KEY (`idequipamento`, `categoria\_idcategoria`),

UNIQUE INDEX `idequipamento\_UNIQUE` (`idequipamento` ASC),

INDEX `fk\_equipamento\_categoria1\_idx` (`categoria\_idcategoria` ASC),

INDEX `fk\_equipamento\_sistemaOperacional1\_idx` (`sistemaOperacional\_idsistemmaOperacional` ASC),

INDEX `fk\_equipamento\_tensao1\_idx` (`tensao\_idtensao` ASC),

INDEX `fk\_equipamento\_corrente1\_idx` (`corrente\_idcorrente` ASC),

INDEX `fk\_equipamento\_ambiente1\_idx` (`ambiente\_idambiente` ASC),

INDEX `fk\_equipamento\_local1\_idx` (`local\_idlocal` ASC),

CONSTRAINT `fk\_equipamento\_categoria1`

FOREIGN KEY (`categoria\_idcategoria`)

REFERENCES `sspmi`.`categoria` (`idcategoria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_equipamento\_sistemaOperacional1`

FOREIGN KEY (`sistemaOperacional\_idsistemmaOperacional`)

REFERENCES `sspmi`.`sistemaOperacional` (`idsistemaOperacional`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_equipamento\_tensao1`

FOREIGN KEY (`tensao\_idtensao`)

REFERENCES `sspmi`.`tensao` (`idtensao`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_equipamento\_corrente1`

FOREIGN KEY (`corrente\_idcorrente`)

REFERENCES `sspmi`.`corrente` (`idcorrente`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_equipamento\_ambiente1`

FOREIGN KEY (`ambiente\_idambiente`)

REFERENCES `sspmi`.`ambiente` (`idambiente`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_equipamento\_local1`

FOREIGN KEY (`local\_idlocal`)

REFERENCES `sspmi`.`local` (`idlocal`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`etapa`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`etapa` (

`idetapa` INT NOT NULL AUTO\_INCREMENT,

`numero` INT NOT NULL,

`dataHoraAbertura` TIMESTAMP NOT NULL,

`dataHoraEncerramento` DATETIME NULL,

`acessoRemoto` VARCHAR(45) NULL,

`descricao` VARCHAR(254) NOT NULL,

`equipamento\_idequipamento` INT NOT NULL,

`protocolo\_idprotocolo` INT NOT NULL,

PRIMARY KEY (`idetapa`, `equipamento\_idequipamento`, `protocolo\_idprotocolo`),

UNIQUE INDEX `idetapa\_UNIQUE` (`idetapa` ASC),

INDEX `fk\_etapa\_equipamento1\_idx` (`equipamento\_idequipamento` ASC),

INDEX `fk\_etapa\_protocolo1\_idx` (`protocolo\_idprotocolo` ASC),

CONSTRAINT `fk\_etapa\_equipamento1`

FOREIGN KEY (`equipamento\_idequipamento`)

REFERENCES `sspmi`.`equipamento` (`idequipamento`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_etapa\_protocolo1`

FOREIGN KEY (`protocolo\_idprotocolo`)

REFERENCES `sspmi`.`protocolo` (`idprotocolo`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`marca`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`marca` (

`idmarca` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

PRIMARY KEY (`idmarca`),

UNIQUE INDEX `idmarca\_UNIQUE` (`idmarca` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`modelo`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`modelo` (

`idmodelo` INT NOT NULL AUTO\_INCREMENT,

`nomenclatura` VARCHAR(100) NOT NULL,

`marca\_idmarca` INT NOT NULL,

`serviceTag` VARCHAR(100) NULL,

`numeroDeSerie` VARCHAR(100) NULL,

PRIMARY KEY (`idmodelo`, `marca\_idmarca`),

UNIQUE INDEX `idmodelo\_UNIQUE` (`idmodelo` ASC),

INDEX `fk\_modelo\_marca1\_idx` (`marca\_idmarca` ASC),

CONSTRAINT `fk\_modelo\_marca1`

FOREIGN KEY (`marca\_idmarca`)

REFERENCES `sspmi`.`marca` (`idmarca`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `sspmi`.`categoria\_has\_marca`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `sspmi`.`categoria\_has\_marca` (

`categoria\_idcategoria` INT NOT NULL,

`marca\_idmarca` INT NULL,

INDEX `fk\_categoria\_has\_marca\_categoria1\_idx` (`categoria\_idcategoria` ASC),

INDEX `fk\_categoria\_has\_marca\_marca1\_idx` (`marca\_idmarca` ASC),

CONSTRAINT `fk\_categoria\_has\_marca\_categoria1`

FOREIGN KEY (`categoria\_idcategoria`)

REFERENCES `sspmi`.`categoria` (`idcategoria`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_categoria\_has\_marca\_marca1`

FOREIGN KEY (`marca\_idmarca`)

REFERENCES `sspmi`.`marca` (`idmarca`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

## CONFIGURAÇÃO E IMPLANTAÇÃO DA APLICAÇÃO WEB

Para essa etapa do projeto foi contatado a Federação de Consórcios, Associações de Municípios de Santa Catarina (FECAM), uma vez que os mesmos são detentores do perfil de nível técnico junto ao Registro.BR do domínio [www.itapoa.sc.gov.br](http://www.itapoa.sc.gov.br). Foi informado via ligação telefônica que os serviços ofertados pela Federação não permitem acesso ao cPanel e Banco de Dados MySql, bem como linguagens PHP 8.0 ou superior.

### SOLUÇÃO PARA HOSPEDAGEM

Contratado serviço de hospedagem da HostBits.  
Custo: R$ 669,60;  
Disco: 10240M;  
Uso de Banda: Ilimitado  
Banco de Dados: MySQL e Postgree  
SSL/ TLS: Sim

## ENTREGAS DO SISTEMA PARA TREINAMENTOS

### 1ª ENTREGA – 1º DE AGOSTO DE 2024

Solicitado aos gestores (Paço Municipal) que indiquem um funcionário que ficará responsável pela solicitação e acompanhamento de sua unidade e no caso da ausência de outro representante, também realizar as solicitações do mesmo. A metodologia adotada facilita a comunicação com apenas alguns funcionários, pois no caso de instabilidade no sistema a demanda é menor.

### 2ª ENTREGA – 1º DE OUTUBRO DE 2024

Solicitado aos gestores (secretarias externas) que indiquem um funcionário que ficará responsável pela solicitação e acompanhamento de sua Secretaria.

#### \_ ENTREGA SAÚDE (PROVISÓRIA) – 13 DE SETEMBRO DE 2024;

\_ ENTREGA CONSELHO TUTELAR – 23 DE SETEMBRO DE 2024;

### 3ª ENTREGA – 14 DE FEVEREIRO DE 2025

Capacitação dos auditores e fiscais da Tributação.

### 4ª ENTREGA – 21 DE FEVEREIRO DE 2025

Capacitação dos funcionários da Tributação.

### 5ª ENTREGA – 28 DE FEVEREIRO DE 2025

Capacitação dos funcionários da Tributação.

### 6ª ENTREGA – 7 DE MARÇO DE 2025

Capacitação dos funcionários da Secretaria de Agricultura e Pesca e Tributação.

### 7ª ENTREGA – 14 DE MARÇO DE 2025

Capacitação dos funcionários do Convênios e Recursos Humanos.

### 8ª ENTREGA – 21 DE MARÇO DE 2025

Capacitação dos funcionários do Frotas, Recursos Humanos e Licitação.

### 9ª ENTREGA – 28 DE MARÇO DE 2025

Capacitação dos funcionários da Licitação.

### 10ª ENTREGA - 04 DE ABRIL DE 2025

Capacitação dos funcionários da Procuradoria Jurídica.