

Sistema German

Manual do Desenvolvedor

GABRIELA BIAR MANERICH

GIANCARLO MAFFEZZOLLI

IAGO SOARES DA SILVA

PLÁCIDO ANTÔNIO LANGER REINERT



Jaraguá do Sul, 2020

# Histórico de Versão

|  |  |  |  |
| --- | --- | --- | --- |
| **DATA** | **VERSÃO** | **DESCRIÇÃO** | **AUTOR** |
| 11/08/2020 | 1.0 | Versão inicial do documento | Plácido |
| 12/12/2020 | 2.0 | Atualizações no documento | Giancarlo |
| 15/12/2020 | 3.0 | Finalização do Documento | Giancarlo/Plácido/Iago |
|  |  |  |  |

# Sumário

[**Histórico de Versão**](#_hqwrbun7agp) **2**

[**Sumário**](#_ywlugxo610kt) **3**

[**1 SOBRE O SISTEMA**](#_yscbiflvrhmy) **4**

[1.1 Sobre o uso do sistema](#_v4kxr1m9jamq) 4

[**2. Arquitetura**](#_lsowk57xhtbd) **5**

[2.1. Diagrama da comunicação Front com API](#_9atuugybi1je) 5

[**3. Codificação Back-End**](#_rr7z4hyt9tsm) **6**

[3.1 UserAction](#_8harcgjkwp1w) 8

[3.3 Action setor (API)](#_ao0ah28tuwwu) 9

[3.4 Action equipamentos(API)](#_6zi657r1e236) 9

[3.5. Action causa de defeito(API)](#_h4i4dxix41nq) 9

[3.6. Action centro de trabalho(API)](#_7ubcxvrrxlq2) 10

[3.7 Ordens de Manutenção](#_kux9qo5ojsw) 10

[3.8 Filtros](#_gwjmldc7mc0l) 11

[**4. Codificação Front-End**](#_2u0avj4501wu) **11**

# 1 SOBRE O SISTEMA

O German é um sistema de gerenciamento das ordens de manutenção, desenvolvido para atuar na empresa de alimentos Duas Rodas.

## 1.1 Sobre o uso do sistema

Inicialmente o sistema recebe as informações do SAP integrado na empresa e com essas informações é gerado um formulário com os dados da manutenção que será preenchido pelo manutentor conforme a manutenção é feita. A partir disso é dado origem a ordem de manutenção digital que é acessível pelos smartphones ou computadores dos funcionários.

## 1.2 Ferramentas Utilizadas Para o desenvolvimento do GERMAN foram utilizados as seguintes ferramentas:

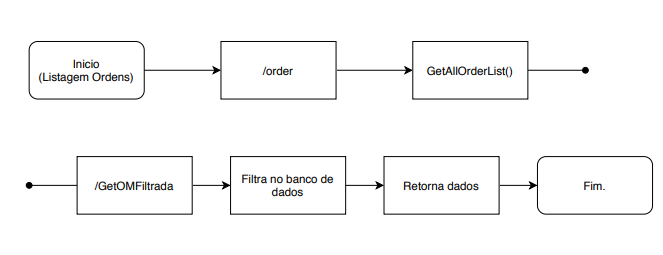
* NodeJs:
  + Link para Download: <https://nodejs.org/en/download/>
  + Documentação: <https://nodejs.org/en/docs/>
* ExpressJs
  + Link explicativo para utilização: <https://expressjs.com/pt-br/starter/installing.html>
  + Documentação: <https://expressjs.com/pt-br/guide/routing.html>
* Vue:
  + Link para instalação: <https://br.vuejs.org/v2/guide/installation.html>
  + Documentação: <https://vuejs.org/v2/guide/>
* Mysql:
  + Link para Download: <https://www.mysql.com/downloads/>
  + Documentação: <https://dev.mysql.com/doc/>

# 2. Arquitetura

## 2.1. Diagrama da comunicação Front com API

O programa possui uma comunicação com o back-end por meio de uma API Rest.

Figura 1: Exemplo de comunicação com API



Na figura 1 está representado o fluxo de processo que o sistema faz para a busca da listagem das ordens de manutenção. Ao logar no sistema ou clicar na tela de lista de ordem de manutenção, o sistema acessa o *endPoint* */order* que por sua vez é responsável por requisitar para o *back-end* a listagem de todas as ordens de manutenção em aberto. Para isso é utilizado um método *GET* chamado *GetAllOrderList()* que vai fazer uma requisição para o *back-end*. Ao iniciar a requisição o back passa por vários arquivos:

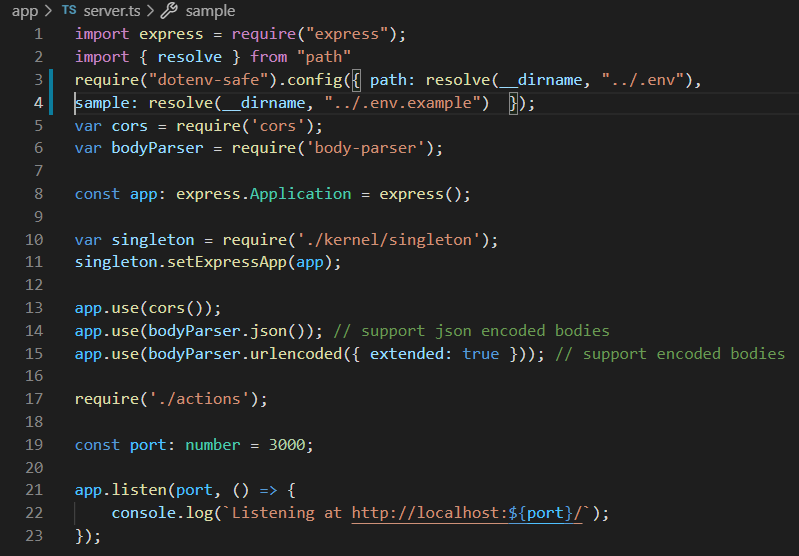
* *Server.ts* : responsável por direcionar a requisição com base na API chamada. Neste caso seria a */GetOMFiltrada*;
* *Filtros.ts* : Possui todas as operações que podem vir a ser chamadas quando é necessário algum filtro na listagem das ordens de manutenção. Também é onde será feita a busca(*select*) no banco de dados para retornar para o *front-end* os dados.

Após esse processo realizado, todas as ordens de manutenção que se encaixam no SQL utilizado são apresentadas em forma de lista no sistema.

# 3. Codificação Back-End

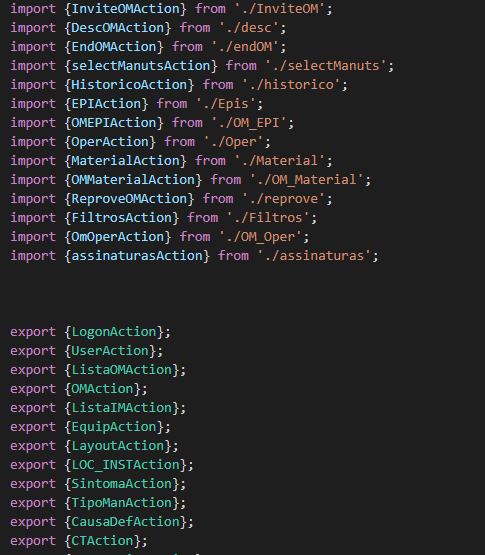
O backend possui um gateway que expõe todos os endpoints da API com base nas classes importadas nele. Cada uma dessas classes, que possuem endpoints, são chamadas de Action. Na figura 2, está uma imagem do arquivo server.ts do German.

Figura 2: server.ts



O server.ts é a porta de entrada para a API, onde fica escutando todos os chamados para a porta especificada e encaminha para o index.ts. Na figura 3 está representado o arquivo index.ts do projeto German.

Figura 3: index.ts



Todos os endpoints seguem uma lógica de retorno que passa por alguns passos exemplificados nos pontos a seguir.

O método exposto pelo /AddUser esta representado na figura 4.

Figura 4: /AddUser

@Post('/AddUser')

public Post(){

this.validateData();

new MySQLFactory().getConnection().select(this.generateADDSQL()).subscribe(

(data : any) => {

if (data.length || data.length > 0){

this.sendError(new KernelUtils().createErrorApiObject(401, '1001', 'Usuário já existe'));

return;

}else{

new MySQLFactory().getConnection().select(this.insertSQL()).subscribe(

(data : any) => {

}

);

}

this.sendAnswer({

token : new VPUtils().generateGUID().toUpperCase()

});

},

(error : any) => {

this.sendError(error);

}

);

}

Inicia pela validação de se possui todos os campos necessários para a ação.  
this.validateData() representado na figura 5.

Figura 5: validação

private validateData(){

new KernelUtils().createExceptionApiError('1001', 'Informe todos os campos corretamente', this.req.body.idsap == null || this.req.body.name == '');

}

Após isso ele faz um select para verificar se existe algum usuário com os dados inseridos no select.

Esse processo é realizado pelo generateADDSQL() apresentado na figura 6.

Figura 6: generateADDSQL()

private generateADDSQL() : string {

return 'select \* from TBUSUARIO where (TBUSUARIO.NOME = \'' + this.req.body.name + '\' OR TBUSUARIO.IDSAP = \'' + this.req.body.idsap + '\') AND STATUS = 1;';

}

E por último, executar o insert. O insert é representado na figura 7.

Figura 7: insertSQL()

private insertSQL() : string{

return 'insert into TBUSUARIO (TBUSUARIO.IDSAP ,TBUSUARIO.LOGIN, TBUSUARIO.SENHA, TBUSUARIO.CARGO, TBUSUARIO.NOME, TBUSUARIO.CDCT) values (\''+ this.req.body.idsap+'\',\''+ this.req.body.login +'\', \''+ this.req.body.password+'\', \''+ this.req.body.cargo+'\', \''+ this.req.body.name+'\', \''+ this.req.body.cdct+'\');';

}

## 3.1 UserAction

Classe: User.ts

POST /AddUser

Body:

{

name:string

idsap:string

login:string

password:string

cargo:string

cdct:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: User.ts

GET /GetUser

Response:

{

ID:int

IDSAP:string

LOGIN:string

SENHA:string

NOME:string

CARGO:int

CDCT:int

STATUS:string

TBCTNOME:string

TBCARGONOME:string

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: User.ts

GET /GetLeads

Response:

{

ID:int

NOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: User.ts

PATCH /DelUser

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: User.ts

POST /EditUser

Body:

{

name:string

Idsap:string

idsaplast:string

login:string

password:string

cargo:int

Cdct:int

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 

## 3.2 Action Ordem de Manutenção

Classe: om.ts

POST /AddOM

Body:

{

titulo:string

idsap:string

solicitante:string

layout:int

ct:int

tipoManut:string

causa:string

def:string

obs:string

prior:string

li:int

requerParada:string

dtIniPlan:string

dtIniProg:string

dtFimPlan:string

dtFimProg:string

Sintoma:int

equip: {

id:int

oper:string

material:string

Qtde:int

}

oper: {

id:int

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: om.ts

Post /DelOM

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: om.ts

POST /EditOM

Body:

{

id:int

titulo:string

idsap:string

solicitante:string

layout:int

ct:int

tipoManut:string

causa:string

def:string

obs:string

prior:string

li:int

requerParada:string

dtIniPlan:string

dtIniProg:string

dtFimPlan:string

dtFimProg:string

Sintoma:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3.3 Action Tipo de manutenção

Classe: TipoMan.ts

POST /AddTIPOMAN

Body:

{

name:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: TipoMan.ts

GET /GetTIPOMAN

Response:

{

ID:int

NOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: TipoMan.ts

Post /DelTIPOMAN

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: TipoMan.ts

POST /EditTIPOMAN

Body:

{

name:string

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Filtros.ts

POST /GetOMFiltrada

Body:

{

GetOMByUserIDFinalizada:string

GetOMsFinalizadaADM:string

GetOMsAndamentoLider:string

GetOMByUserIDFinalizada:string

GetOMsAndamentoADM:string

GetOMsFinalizadaLider:string

GetOMsBySetor:string

GetOM:string

GetOMByUserIDAtribuida:string

id:string

estado:string

prioridade:string

locinst:string

idct:string

idlayout:string

tpom:string

sintoma:string

causadef:string

requerparada:string

idsap:string

geracaoMin:string

geracaoMax:string

iniPlanMin:string

iniPlanMax:string

iniProgMin:string

iniProgMax:string

fimPlanMin:string

fimPlanMax:string

fimProgMin:string

fimProgMax:string

orderbydesc:string

orderbyasc:string

}

Response:

{

SINTOMANOME:string

TBLOC\_INSTNOME:string

TBCTNOME:string

TBLAYOUTOMNOME:string

ESTILO:string

TBTIPOMANNOME:string

TBCAUSADEFNOME:string

TBPRIORIDADENOME:string

ID:string

IDSAP:string

TITULO:string

SOLIC:string

IDLAYOUT:string

IDCT:string

TPOM:string

SINTOMA:string

CAUSADEF:string

PRIORIDADE:string

DEF:string

DTGERACAO:string

DTBAIXA\_MANUT:string

DTBAIXA\_SETOR:string

DTBAIXA\_ADMIN:string

DT\_INI\_PLAN:string

DT\_INI\_PROG:string

DT\_FIM\_PLAN:string

DT\_FIM\_PROG:string

OBS:string

ESTADO:string

LOC\_INST\_ATRIB:string

STATUS:string

ISREJECTED:string

REQUERPARADA:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3.4 Action Sintoma

Classe: Sintoma.ts

POST /AddSINTOMA

Body:

{

name:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Classe: Sintoma.ts

GET /GetSINTOMA

Response:

{

ID:int

NOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Sintoma.ts

PATCH /DelSINTOMA

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Sintoma.ts

POST /EditSINTOMA

Body:

{

name:string

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3.5. Action causa de defeito

Classe: CausaDef.ts

POST /AddCAUSADEF

Body:

{

name:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Classe: CausaDef.ts

GET /GetCAUSADEF

Response:

{

ID:int

NOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: CausaDef.ts

PATCH /DelCAUSADEF

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: CausaDef.ts

POST /EditCAUSADEF

Body:

{

name:string

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3.6. Action centro de trabalho

Classe: CT.ts

POST /AddCT

Body:

{

name:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Classe: CT.ts

GET /GetCT

Response:

{

ID:int

NOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: CT.ts

PATCH /DelCT

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: CT.ts

POST /EditCT

Body:

{

name:string

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3.7 Action Operações

Classe: Oper.ts

POST /AddOPER

Body:

{

desc:string

idsap:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Oper.ts

GET /GetOPER

Response:

{

ID:int

NOME:string

IDSAP:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Oper.ts

PATCH /DelOPER

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3.8. Action Layout

Classe: Layout.ts

POST /AddLAYOUTOM

Body:

{

name:string

layout:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Classe: Layout.ts

GET /GetLAYOUTOM

Response:

{

ID:int

NOME:string

IDESTILO:int

STATUS:string

ESTILONOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Layout.ts

PATCH /DelLAYOUTOM

Body:

{

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Classe: Layout.ts

POST /EditLAYOUTOM

Body:

{

name:string

layout:int

id:int

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Classe: Layout.ts

GET /GetESTILO

Response:

{

ID:int

NOME:string

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 4. Codificação Front-End

Exemplo básico da linguagem VueJS apresentado na figura 8:

Figura 8: Código básico VueJs.



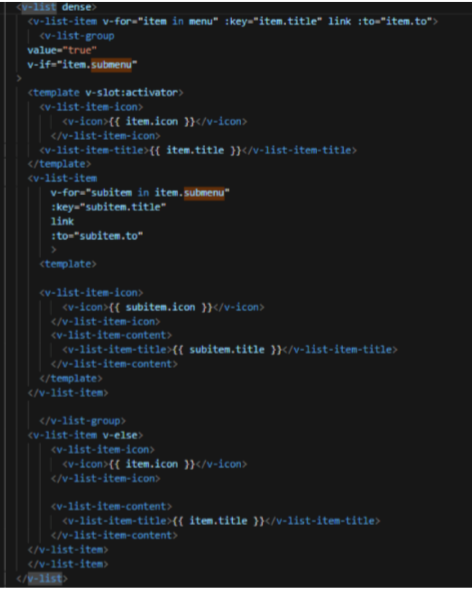
Em uma aplicação em VueJS é necessário ter alguns principais documentos, nesta seção será demonstrado os principais documentos no projeto German.

Iniciando com o "app.vue", que é o arquivo principal do front-end, onde o componente que for inserido nele, vai aparecer em todo o sistema por exemplo o componente:

* <v-list> </v-list> Que é utilizado para criar o menu lateral que aparece em todas as telas do sistema.

Na ﬁgura 9 é possível observar um exemplo do componente acima.

Figura 9: V-List no VueJs.

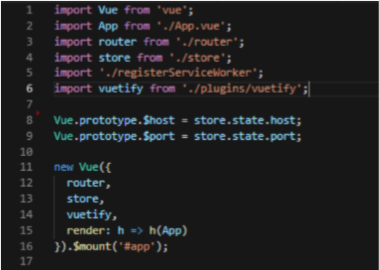


Observando a ﬁgura 10 é possível observar o arquivo "main.js".

Responsável por iniciar a aplicação, quando o sistema for iniciado é esse o arquivo que vai rodar primeiro, levantando todo o sistema.

Como por exemplo na ﬁgura 10, onde foi importado o router que é responsável pelas rotas do sistema e quando o New Vue é iniciado, ele chama as rotas.

Figura 10: Exemplo main.js



Por último o "Router.js". Responsável pelo roteamento do sistema, aqui é feito toda a conﬁguração das telas e de suas url’s de acesso, por exemplo, como mostrado na ﬁgura 26.

Visto isso é necessário informar ao sistemas as rotas na listagem de rotas do sistema, que ﬁca dentro da lista routes[] que são nomeadas, informando, nas conﬁgurações do arquivo os dados:

* Patch(caminho): url.
* Name: nome da rota.

Desta forma o sistema entende que ao direcionar a url, ele deve respeitar seu nome e apresentar a tela que foi conﬁgurada para aquele nome. Assim o sistema consegue de uma forma simples e organizada, mapear todas as telas existentes no German. O arquivo "Router.js"do German esta representado na ﬁgura 11.

Figura 11: Exemplo Router.js

Figura 11: Exemplo Router.js

