MED-SYSTEM

# Informática de Gestão

## Relatório Final

2020/2021



Docente: Miguel Bugalho

Duarte Nuno Carvalho Lobato – 50034922, 3º Ano, 1º Semestre

# **Índice**

Conteúdo

[**Índice** 2](#_Toc63351406)

[**Enquadramento** 3](#_Toc63351407)

[**Cenários/ Casos de Utilização** 4](#_Toc63351408)

[Principal 4](#_Toc63351409)

[Secundários 4](#_Toc63351410)

[**Modelo de Domínio** 6](#_Toc63351411)

[**Requisitos Funcionais** 7](#_Toc63351412)

[**Documentação REST** 8](#_Toc63351413)

[**Guião** 11](#_Toc63351414)

[**Mockups vs Screenshots** 11](#_Toc63351415)

[**Conclusão** 14](#_Toc63351416)

# **Enquadramento**

Este projeto é desenvolvido no âmbito da cadeira de Programação WEB, na qual nos é imposto o desafio de criar uma aplicação completamente funcional de forma a utilizarmos conhecimentos e ferramentas previamente adquiridos e também adquiridos ao longo do semestre. No seu desenvolvimento, foram usadas as linguagens HTML, CSS e JavaScript, foram ainda usadas ferramentas como o Node.js e frameworks como o Express.js.

Embora a minha ideia inicial passasse por pegar numa ideia de um projeto anteriormente concebido, decidi começar do zero e construir um novo plano estruturado para a criação de uma ideia melhor e de um projeto mais rico e complexo. Desta forma surge o MED-SYSTEM, MED-SYSTEM é uma aplicação que se insere na área de saúde, designadamente no ambiente de sistemas de internamento hospitalar. Esta aplicação é uma ferramenta de auxílio para a gestão de sistemas hospitalares, foca-se portanto na designação das camas indicadas para cada paciente e transmite informações ao utilizador sobre cada paciente e sobre o sistema em si.

# **Cenários/ Casos de Utilização**

# Principal

1. VISUALIZAÇÃO DE TODAS AS ALAS/CAMAS/PACIENTES

O utilizador entra na aplicação > clica nas alas > a interface apresenta as alas hospitalares existentes no hospital > seleciona-se a ala > é possível a visualização de todas as camas dessa ala através de círculos correspondentes às mesmas e cores que representam o estado dos pacientes que as ocupam, caso estejam vazias não apresentam cor > o utilizador clica numa bola que corresponda a uma cama ocupada > popup é aberto > toda a informação sobre o ocupante dessa cama pode ser visualizada

# Secundários

1. CRIAÇÂO DE FICHA MÉDICA

O utilizador entra na aplicação > clica em “Gestão Hospitalar” > é imediatamente aberto o separador para criação de novo paciente > preenche os dados pedidos > clica “Inserir” > um novo paciente foi adicionado à base de dados

1. ATRIBUIÇÃO DE CAMA

O utilizador entra na aplicação > clica em " Gestão Hospitalar” > clica no separador “Cama” > seleciona os dados pedidos > clica “Atribuir” > paciente é atribuído a cama

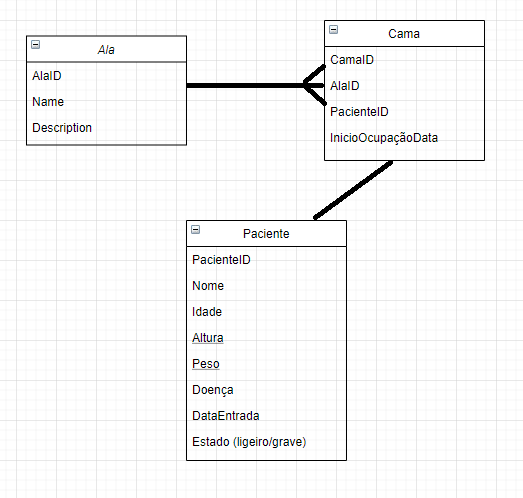
1. REMOÇÃO DE PACIENTE

O utilizador entra na aplicação > clica em " Gestão Hospitalar” > clica no separador “Gestão” > visualiza uma tabela onde são demonstradas as camas ocupadas e o nome dos pacientes que as ocupam > clica no botão do paciente que quer remover > a cama fica vazia e o paciente é apagado da base de dados

1. VISUALIZAÇÃO DE INDICADORES

O utilizador entra na aplicação > clica em “Indicadores” > é redirecionado para uma página onde pode visualizar informações relativas ao hospital, como o número de camas, número de pacientes internados, …

# **Modelo de Domínio**



# **Requisitos Funcionais**

* Visualizar alas, camas e pacientes
* Criar Ficha médica
* Atribuir cama
* Remover paciente da respetiva cama
* Visualizar indicadores
* Criar popup de paciente
* Armazenar dados de alas, camas e pacientes
* Criar tabela dinâmica/funcional para a remoção de pacientes

# **Documentação REST**

Recurso Camas  ( /api/camas )

|  |
| --- |
| **Listar todas as camas vazias**  Devolve todas as camas onde não estão inseridos pacientes (cama\_pac\_id >0), isto impede que seja atribuída a um novo paciente, uma cama ocupada. |
| /api/camas  **(get)** |
| Sucesso(200):   |  | | --- | | [{"cama\_id":6,"cama\_ala\_id":2,"cama\_pac\_id":0,"cama\_datainicial":"2020-12-01T00:00:00.000Z"},{"cama\_id":8,"cama\_ala\_id":2,"cama\_pac\_id":0,"cama\_datainicial":"1900-01-01T00:36:45.000Z"},{"cama\_id":9,"cama\_ala\_id":2,"cama\_pac\_id":0,"cama\_datainicial":"1900-01-01T00:36:45.000Z"}  ] | |
| Erro:  500: Erro do servidor |
| Exemplo:  let bedElem = document.getElementById("bed");    try {        let camas = await $.ajax({            url: "http://localhost:3001/api/camas",            method: "get",            dataType: "json"        }); |

|  |
| --- |
| **Atualizar uma cama vazia que recebeu um doente**  Recebe o valor do id do paciente que irá ser colocado na cama e o valor do id da cama onde será colocado, atualiza o valor de cama\_pac\_id e a cama passa a estar ocupada |
| /api/camas  **(put)** |
| Dados:  {  } |
| Sucesso(200):  {  fieldCount: 0,  affectedRows: 0,  insertId: 0,  serverStatus: 2,  warningCount: 0,  message: '',  protocol41: true,  changedRows: 0 } |
| Erro:  500: Erro do servidor |
| Exemplo:   let result2 = await $.ajax({                      url:"http://localhost:3001/api/camas",                      method:"put",                      data: JSON.stringify(cama),                      contentType: "application/json",                      dataType:"json"                  }); |

|  |
| --- |
| **Listar todas as camas preenchidas**  Devolve o id de todas as camas preenchidas (cama\_pac\_id >0) e o nome de cada paciente que as ocupa, numa lista organizada por ordem cresecente de id |
| /api/camas/rows  **(get)** |
| Sucesso(200):  {  [{"cama\_id":1,"pac\_nome":"João"},{"cama\_id":2,"pac\_nome":"test"},  {"cama\_id":3,"pac\_nome":"Maria"},{"cama\_id":4,"pac\_nome":"Pedro"},  {"cama\_id":5,"pac\_nome":"asd"},{"cama\_id":7,"pac\_nome":"test"}  ]  } |
| Erro:  500: Erro do servidor |
| Exemplo:  let result = await $.ajax({            url: "http://localhost:3001/api/camas/rows",            method: "get",            dataType: "json"        }); |

|  |
| --- |
| **Devolver o número de camas existentes, ocupadas e livres**  Devolve uma contagem de camas existentes, ocupadas e livres |
| /api/camas/ind  **(get)** |
| Sucesso(200):  [{"count1":20,"count2":14,"count3":6}] |
| Erro:  500: Erro do servidor |
| Exemplo:  let ind = await $.ajax({              url: "http://localhost:3001/api/camas/ind",              method: "get",              dataType: "json" |

|  |
| --- |
| **Atualizar uma cama que passou a estar vazia**  Devolve o valor do id do paciente recebido, que irá ser removido e atualiza o valor de cama\_pac\_id onde esse paciente se encontrava, tornando a cama vazia |
| /api/camas/:cama\_id  **(put)** |
| Parâmetros:  Id(obrigatório): id da cama ocupada, da qual o paciente será removido |
| Dados:  {  cama\_id: 1  } |
| Sucesso(200):  {fieldCount: 0,  affectedRows: 0,  insertId: 0,  serverStatus: 2,  warningCount: 0,  message: '',  protocol41: true,  changedRows: 0 }} |
| Erro:  500: Erro do servidor |
| Exemplo:  let result = await $.ajax({              url:"http://localhost:3001/api/camas/"+cama\_id,              method:"put",              contentType: "application/json",              dataType:"json"          }); |

Recurso Estados

|  |
| --- |
| **Listar todas as camas e todos os estados**  Devolve uma lista com os ids de todas as camas existentes, respetivas alas e respetivos estados dos pacientes presentes nessas camas |
| /api/estados  **(get)** |
| Sucesso(200):  [{"ala\_id":1,"cama\_id":1,"pac\_estado":"ESTAVEL"},  {"ala\_id":1,"cama\_id":2,"pac\_estado":"ESTAVEL"},  {"ala\_id":1,"cama\_id":3,"pac\_estado":"ALTA"},  {"ala\_id":1,"cama\_id":4,"pac\_estado":"GRAVE"},  {"ala\_id":1,"cama\_id":5,"pac\_estado":"nada"},  {"ala\_id":2,"cama\_id":6,"pac\_estado":"nada"},  {"ala\_id":2,"cama\_id":7,"pac\_estado":"ESTAVEL"},  }] |
| Erro:  500: Erro do servidor |
| Exemplo:  let estados = await $.ajax({              url: "http://localhost:3001/api/estados",              method: "get",              dataType: "json" |

Recurso Pacientes

|  |
| --- |
| **Listar todos os pacientes**  Devolve uma lista com todos os pacientes existentes na base de dados e os seus dados respetivos(nome, idade, altura, peso, doença, data de internamento e estado) |
| /api/pacientes  **(get)** |
| Sucesso(200):  [{"pac\_id":4,"pac\_nome":"Pedro","pac\_idade":76,"pac\_altura":1.78,  "pac\_peso":70.74,"pac\_doenca":"Cancro","pac\_entrada":"2020-12-01T00:00:00.000Z","pac\_estado":"GRAVE"}] |
| Erro: |
| Exemplo:  let pacientes = await $.ajax({            url: "http://localhost:3001/api/pacientes",            method: "get",            dataType: "json" |

|  |
| --- |
| **Inserir uma ficha médica na base de dados**  Recebe as informações necessárias para preencher a ficha médica de um novo paciente e esse paciente é inserido na base de dados |
| /api/pacientes  **(post)** |
|  |
| Sucesso(200):  {  fieldCount: 1,  affectedRows: 1,  insertId: 1,  serverStatus: 2,  warningCount: 0,  message: '',  protocol41: true,  changedRows: 0 }  } |
| Erro:  500: Erro do servidor |
| Exemplo:  let result = await $.ajax({              url:"http://localhost:3001/api/pacientes",              method:"post",              data: JSON.stringify(paciente),              contentType: "application/json",              dataType:"json"          }); |

|  |
| --- |
| **Devolver o número de pacientes com alta, estáveis e graves**  Devolve uma contagem de pacientes com alta, estáveis e graves |
| /api/pacientes/ind  **(get)** |
| Sucesso(200):  {  [{"count1":1,"count2":2,"count3":1}]  } |
| Erro:  500: Erro do servidor |
| Exemplo:   let ind = await $.ajax({              url: "http://localhost:3001/api/pacientes/ind",              method: "get",              dataType: "json"          }); |

|  |
| --- |
| **Listar estados**  Faz um distinct e devolve os valores dos estados existentes na base de dados |
| /api/pacientes/estados  **(get)** |
| Sucesso(200):  [{"pac\_estado":"nada"},{"pac\_estado":"ESTAVEL"},  {"pac\_estado":"ALTA"},{"pac\_estado":"GRAVE"}] |
| Erro:  500: Erro do servidor |
| Exemplo:  let stateElem = document.getElementById("state");    try {        let distest = await $.ajax({            url: "http://localhost:3001/api/pacientes/estados",            method: "get",            dataType: "json"        }); |

|  |
| --- |
| **Devolver os dados de um paciente numa cama**  Devolve toda a informação do paciente na cama do cama\_id recebido |
| /api/pacientes/:cama\_id  **(get)** |
| Parâmetros:  Id(obrigatório); id do paciente do qual queremos informação |
| Sucesso(200):  {  [{pac\_nome:”Maria”, pac\_id:1, pac\_idade:89, pac\_altura:1.65, pac\_peso: 50.1, pac\_doenca: “Fratura”, paca\_entrada: 2020-12-06, pac\_estado: “ALTA”, cama\_id: 3}]  } |
| Erro:  500: Erro do servidor |
| Exemplo:  let ficha = await $.ajax({              url:"http://localhost:3001/api/pacientes/"+cama\_id,              method:"get",              dataType:"json" |

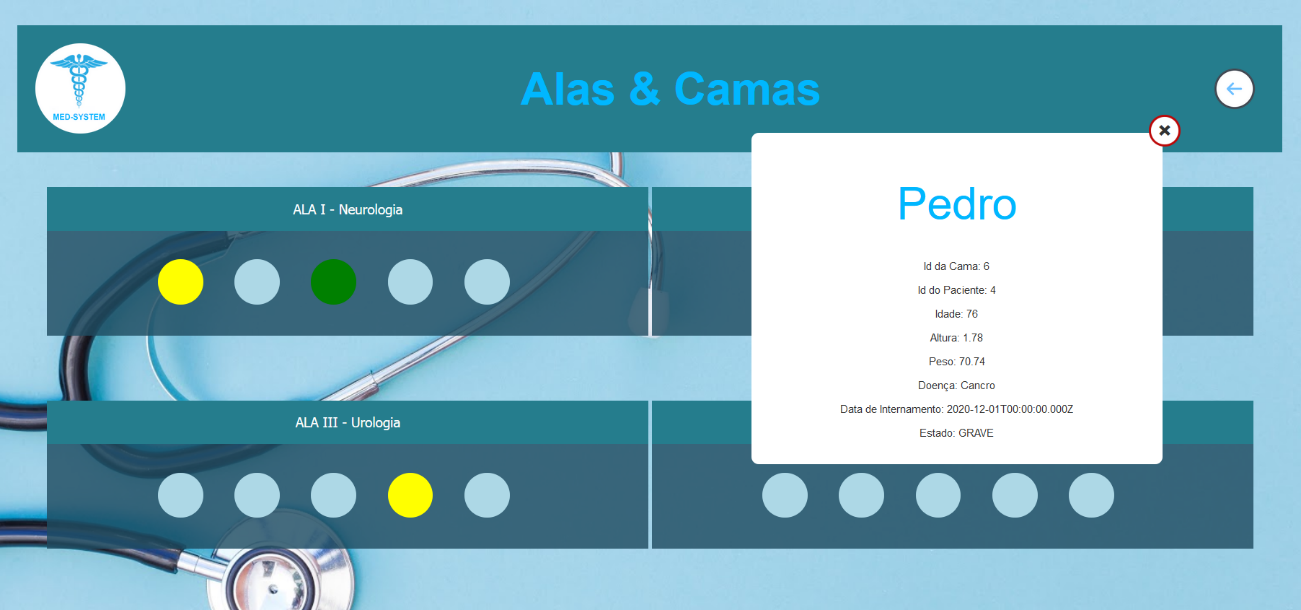
|  |
| --- |
| **Remover um paciente**  Pega no valor do id do paciente devolvido em /api/camas/:cama:id (put), cuja cama onde estava já se encontra vazia, e apaga esse paciente da base de dados |
| /api/camas/:pac\_id  **(delete)** |
| Parâmetros:  Id(obrigatório): o id do paciente a remover |
| Dados:  {pac\_id: 1} |
| fieldCount: 0,  affectedRows: 0,  insertId: 0,  serverStatus: 2,  warningCount: 0,  message: '',  protocol41: true,  changedRows: 0 } |
| Erro:  500: Erro do servidor |
| Exemplo:   let result2 = await $.ajax({              url:"http://localhost:3001/api/pacientes/"+result.data[0].pac\_id,              method:"delete",              contentType: "application/json",              dataType:"json"          }); |

# **Guião**

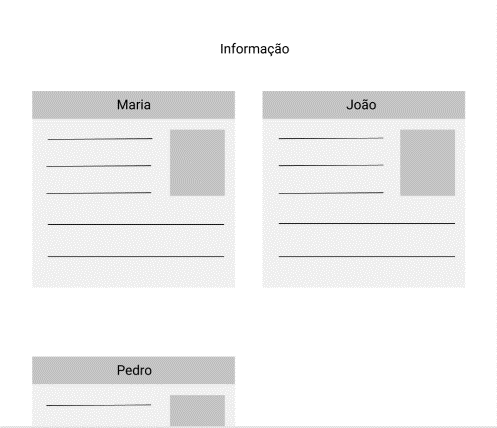
# **Mockups vs Screenshots**

1. VISUALIZAÇÃO DE TODAS AS ALAS/CAMAS/PACIENTES

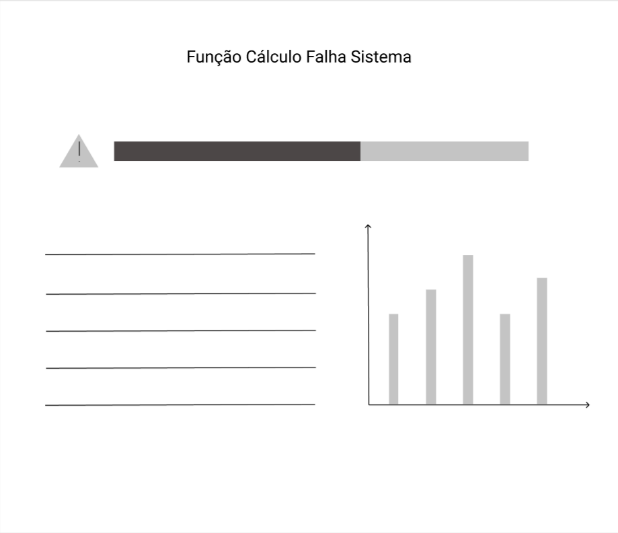




1. CRIAÇÂO DE FICHA MÉDICA
2. ATRIBUIÇÃO DE CAMA
3. REMOÇÃO DE PACIENTE



1. 5VISUALIZAÇÃO DE INDICADORES



# **Conclusão**

Pontos importantes a retirar do projeto MED-SYSTEM:

* Bom tema de projeto
* Melhoria significativa em relação à proposta;
* Desenvolvimento notável, a nível pessoal, de skills nas linguagens usadas;
* Melhoria no conteúdo/organização dos pdfs da cadeira facilitaram bastante todo o percurso feito durante o desenvolvimento do projeto.