

Centro Universitário Senac

Gustavo Orlando Araújo Vergani

Jackson Henrique Ferreira

João Victor Farias Teixeira

Lucas da Silva Macedo

Maria Mercedes da Silva Rodrigues

**PROJETO INTEGRADOR 2:**

MONITORAMENTO DE VARIAÇÃO DE NÍVEL DE ÁGUA

São Paulo

2023

Gustavo Orlando Araújo Vergani

Jackson Henrique Ferreira

João Victor Farias Teixeira

Lucas da Silva Macedo

Maria Mercedes da Silva Rodrigues

## **PROJETO INTEGRADOR 2:**

### **MONITORAMENTO DE VARIAÇÃO DE NÍVEL DE ÁGUA**

Projeto Integrador apresentado ao curso de Análise e Desenvolvimento de Sistemas, do Centro Universitário Senac – Santo Amaro, como requisito para obtenção de aprovação em disciplina.

Orientadores: Alexandre Igosheff e Evandro Teruel.

São Paulo

2023

## ÍNDICE DE ILUSTRAÇÕES

FIGURA 1: MONTAGEM APRESENTAÇÃO 2 .....	14
FIGURA 2: CIRCUITO.....	14
FIGURA 3: CIRCUITO.....	14
FIGURA 4: CIRCUITO.....	15
FIGURA 5: DISPLAY .....	15
FIGURA 6: HOME .....	27
FIGURA 7: TELA "ÚLTIMA ATUALIZAÇÃO" .....	27
FIGURA 8: TELA "HISTÓRICO DE ATUALIZAÇÕES" .....	28
FIGURA 9: TELA "REMOÇÃO DE DADOS" .....	28
FIGURA 10: CÓDIGO HTML .....	29
FIGURA 11: EXCLUSÃO DE DADOS HTML.....	29
FIGURA 12: EXCLUSÃO DE DADOS JSP 1 .....	30
FIGURA 13: EXCLUSÃO DE DADOS JSP 2 .....	30
FIGURA 14: ÚLTIMA ATUALIZAÇÃO JSP 1 .....	31
FIGURA 15: ÚLTIMA ATUALIZAÇÃO JSP 2 .....	31
FIGURA 16: HISTÓRICO DE DADOS JSP 1 .....	32
FIGURA 17: HISTÓRICO DE DADOS JSP 2 .....	32

## ÍNDICE DE TABELAS

<b>TABELA 1: SENSOR BOIA HORIZONTAL .....</b>	<b>8</b>
<b>TABELA 2: ESP8266.....</b>	<b>9</b>
<b>TABELA 3: PROTOBOARD .....</b>	<b>10</b>
<b>TABELA 4: JUMPERS .....</b>	<b>10</b>
<b>TABELA 5: LEDS .....</b>	<b>10</b>
<b>TABELA 6: RESISTOR .....</b>	<b>10</b>
<b>TABELA 7: SUQUEIRA .....</b>	<b>11</b>
<b>TABELA 8: DISPLAY .....</b>	<b>11</b>
<b>TABELA 9: BUZZER.....</b>	<b>11</b>

## **LISTA DE ABREVIATURAS E SIGLAS**

IoT – Internet of Things (Internet das Coisas).

LED – Light Emitting Diode (Diodo Emissor de Luz).

SQL – Structured Query Language (Linguagem de Consulta Estruturada).

BD – Banco de Dados.

CRUD – Create, Read, Update, Delete (Criar, Ler, Atualizar, Deletar).

WEB – World Electronic Base.

WI-FI – Wireless Fidelity.

## SUMÁRIO

<b>INTRODUÇÃO .....</b>	<b>7</b>
<b>HARDWARE E PROGRAMAÇÃO .....</b>	<b>8</b>
<b>STREAMING DE DADOS.....</b>	<b>12</b>
<b>CONSTRUÇÃO DO SISTEMA .....</b>	<b>12</b>
<b>MONTAGEM DO SISTEMA.....</b>	<b>13</b>
<b>CÓDIGO FONTE.....</b>	<b>15</b>
<b>APLICAÇÃO WEB .....</b>	<b>26</b>
<b>CÓDIGO DOS ARQUIVOS WEB .....</b>	<b>28</b>
<b>CÓDIGO CSS .....</b>	<b>33</b>
.....	37
.....	38
<b>CONCLUSÃO .....</b>	<b>39</b>
<b>REFERÊNCIAS .....</b>	<b>40</b>

## **INTRODUÇÃO**

Este trabalho tem o objetivo de apresentar o protótipo final do projeto de IoT, referente à última apresentação da disciplina de Projeto Integrador. Nós desenvolvemos um protótipo de sensor de nível para detectar se um reservatório está com água entre cheio e vazio.

O NodeMCU ESP-8266 foi o microcontrolador escolhido para realizar a automatização do controle hídrico. Ele atuará em conjunto com os componentes definidos pelo grupo. Pela sua capacidade de conexão com o WI-FI, encaminharemos as medições obtidas diretamente para uma base de dados. Essas informações serão exibidas através de uma aplicação Web.

**HARDWARE E PROGRAMAÇÃO**

Listagem das especificações técnicas de cada produto utilizado até o momento.

**Sensor Nível de Água - Boia Horizontal**

<b>Modelo</b>	<b>CS-C0058</b>
<b>Marca</b>	OEM
<b>Tensão Máxima de Contato</b>	220V DC/AC
<b>Classificação Máxima de Contato</b>	10W
<b>Corrente de Comutação Máxima</b>	0.5A
<b>Tensão de Ruptura Máxima</b>	100V DC/AC
<b>Corrente Máxima</b>	1.0A
<b>Resistência Máxima de Contato</b>	100m Ohms
<b>Faixa de Temperatura</b>	-10 ~ +60°
<b>Material do Flutuador</b>	P.P
<b>Comprimento do Cabo</b>	36 cm
<b>Composição</b>	Plástico, Metal
<b>Tamanho</b>	56mm Largura x 24mm de Comprimento x 24mm de Altura
<b>Peso</b>	10 g

Tabela 1: Sensor Boia Horizontal



## ESP8266 NodeMCU

Módulo NodeMcu Lua ESP-12E
Versão do módulo: V2
Memória flash: 4 MB
Tensão de operação: <ul style="list-style-type: none"><li>• Pinos Digitais: 3,3 V</li><li>• Pino Analógico: 1,0 V</li></ul>
Wireless padrão 802.11 b/g/n
Antena embutida
Conector micro-usb para programação e alimentação
Modos de operação: STA/AP/STA+AP
Suporta 5 conexões TCP/IP
Portas GPIO: 13
GPIO com funções de PWM, I2C, SPI, etc.
Resolução do PWM: 10 bits (valores de 0 a 1023)
01x conversor analógico digital (ADC)
Distância entre pinos: 2,54 mm
Dimensões: 49 x 26 x 7 mm (sem considerar os pinos)

Tabela 2: ESP8266

**Protoboard**

<b>Furos</b>	400
<b>Material</b>	Plástico ABS
<b>Resistência de Isolamento</b>	100MO min
<b>Tensão Máxima</b>	500v AC por minuto
<b>Faixa de Temperatura</b>	- 20 a 80°C
<b>Dimensões</b>	8,3 x 5,5 x 1,0 cm
<b>Para terminais e condutores de 0,3 a 0,8 mm (20 a 29 AWG)</b>	

Tabela 3: Protoboard

**Jumpers**

<b>Tipo</b>	Macho/Macho
<b>Comprimento</b>	10cm

Tabela 4: Jumpers

**LEDs**

<b>Cores</b>	Verde, Amarelo e Vermelho.
--------------	----------------------------

Tabela 5: LEDs

**Resistor**

<b>Valor</b>	220 Ohms
<b>Tolerância</b>	5%
<b>Potência</b>	1/4W

Tabela 6: Resistor

**Recipiente**

<b>Litragem</b>	3 litros e 600 mililitros
-----------------	---------------------------

Tabela 7: Suqueira

#### Display

<b>Cor da luz de fundo</b>	Azul
<b>Quantidade de dígitos</b>	8192
<b>Quantidade de linhas</b>	64
<b>Altura total</b>	3 mm
<b>Comprimento total</b>	30 mm
<b>Largura total</b>	27 mm

Tabela 8: Display

#### Buzzer Ativo

<b>Tensão de Operação</b>	3,5 – 5 v
<b>Tipo</b>	Beep contínuo
<b>Cor</b>	Preto
<b>Diâmetro</b>	12mm
<b>Altura</b>	10mm

Tabela 9: Buzzer

## **STREAMING DE DADOS**

Utilizamos a biblioteca `MySQL_MariaDB_Generic` para realizar o streaming de dados entre o ESP8266 e o banco de dados. O BD escolhido foi o “db4free”, um banco de dados online que oferece um serviço para testes com as versões mais recentes – e em desenvolvimento – do MySQL Server. É necessário criar uma conta, gratuitamente, para criar e testar as aplicações

Para linká-lo ao código, todas as credenciais do BD (endereço do servidor, o nome de usuário e a senha, e o nome do BD) estão declaradas no código fonte. A partir desses dados, a placa, conectada a uma rede WI-FI, consegue se comunicar com o BD.

## **CONSTRUÇÃO DO SISTEMA**

O fio GND é conectado no polo negativo da protoboard, o qual se conecta aos polos negativos da Boia, e dos LEDs juntamente com seus resistores.

Temos os pinos de entrada dos sensores, sendo eles: o pino D1 e o pino D2, que estarão ligados

ao polo positivo da protoboard. O pino de distribuição 3v também estará conectado ao polo positivo da protoboard.

Como saída, temos o pino D3, D4 e D5, os quais após receber o estado do sensor, irão acender de acordo com o nível de água presente.

Adicionamos o Buzzer no pino D6, e o Display está nos pinos D7 e D8, que informam e recebem o texto, a imagem e seus dados.

## **MONTAGEM DO SISTEMA**



Figura 1: Montagem Apresentação 2

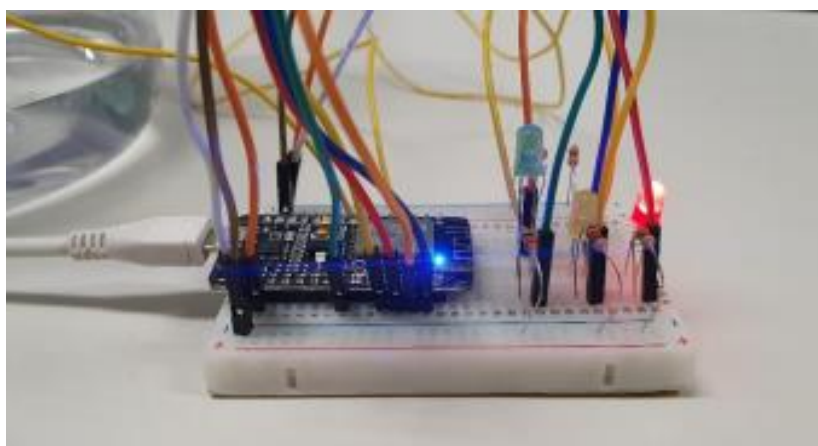


Figura 2: Circuito

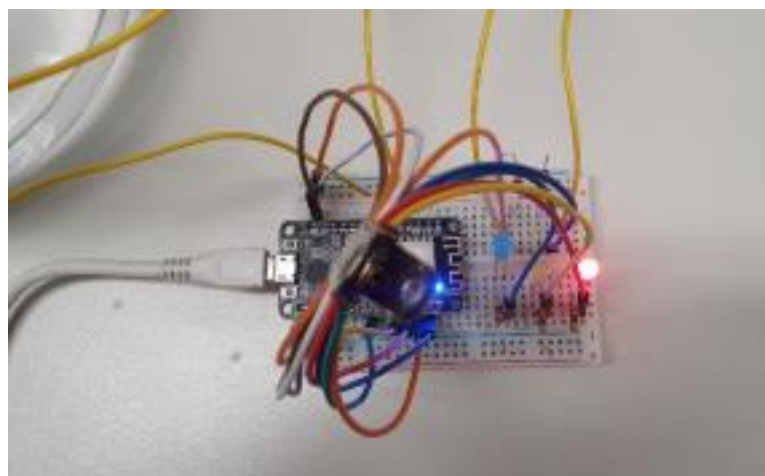


Figura 3: Circuito

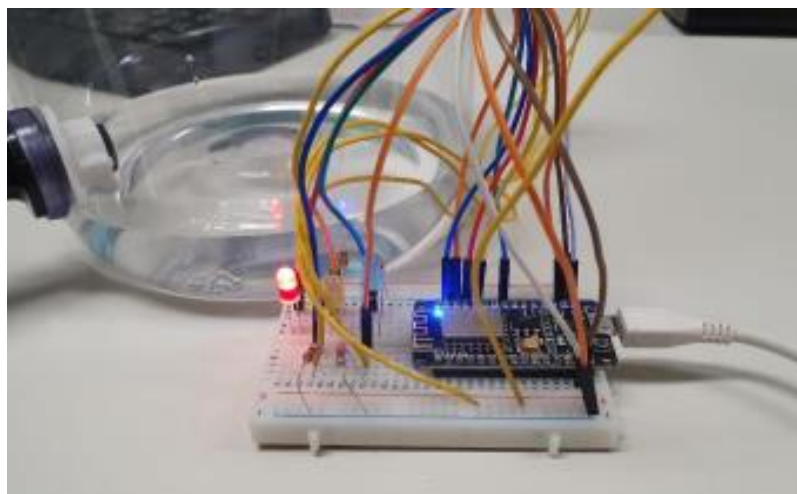


Figura 4: Circuito



Figura 5: Display

**CÓDIGO FONTE**

Apresentação do código fonte em linguagem C, desenvolvido para a automação do sistema projetado. O código já contém seus devidos comentários.

```
// inclui biblioteca do display
```

```
#include <Wire.h>
```

```
#include "SSD1306Wire.h"
```

```
// inclue as bibliotecas de wi-fi e conexão com o mySQL
```

```
#include <MySQL_Generic.h>
```

```
#include <MySQL_Generic.hpp>
```

```
#include <ESP8266WiFi.h>
```

```
// define pinagem
```

```
#define sensorTopo D1
```

```
#define sensorBaixo D2
```

```
#define LED_1 D3
```

```
#define LED_2 D4
```

```
#define LED_3 D5
```

```
#define BUZZER D6
```

```
// define variáveis para guardar as condições dos sensores
```



```
bool sensorTopoAtivo = false;
```

```
bool sensorBaixoAtivo = false;
```

```
bool ultimoSensorTopoAtivo = false;
```

```
bool ultimoSensorBaixoAtivo = false;
```

```
int tempo = 400;
```

```
SSD1306Wire display(0x3c, D7, D8);
```

```
// define as credenciais para a conexão com o wi-fi
```

```
const char* redeWIFI = "JV";
```

```
const char* senhaWIFI = "@Victor161931";
```

```
// define as credenciais para a conexão com o banco de dados
```

```
IPAddress servidor (85, 10, 205, 173);
```

```
char usuario[] = "grupopi";
```

```
char senha[] = "gjjlm123";
```

```
char bd[] = "reservatorio";
```

```
// cria os objetos de conexão e envio de infos para o banco de dados
```

```
MySQL_Connection conn((Client *)&client);
```

```
MySQL_Query *query_mem;
```

```
void setup() {
```

```
    //Inicia o monitor serial
```

```
    Serial.begin(9600);
```

```
    // inicia conexão com o wifi
```

```
    WiFi.begin(redeWIFI, senhaWIFI);
```

```
    // tenta conectar ao wi-fi enquanto a conexão não for concluída
```

```
    while (WiFi.status() != WL_CONNECTED) {
```

```
        delay(1000);
```

```
        Serial.println("Conectando ao WIFI...");
```

```
    }
```

```
    Serial.println("WIFI conectado!");
```

```
    // tenta conectar ao banco de dados enquanto a conexão não for concluída
```

```
while (!conn.connect(servidor, 3306, usuario, senha, bd)) {

    Serial.println("Conectando ao Banco de Dados.");

}

Serial.println("Conexão com o Banco de Dados concluída.");

// configura pinos de entrada e saída

pinMode(sensorTopo, INPUT);

pinMode(sensorBaixo, INPUT);

pinMode(LED_1, OUTPUT);

pinMode(LED_2, OUTPUT);

pinMode(LED_3, OUTPUT);

pinMode(BUZZER, OUTPUT);

Serial.println("Estado das boias");

Serial.println();
```

```
//Inicia o display
```

```
Serial.begin(115200);

display.init();

display.flipScreenVertically();

}

void loop() {

    // atribui valor as variáveis que guardam o estado dos sensores

    sensorTopoAtivo = digitalRead(sensorTopo) == HIGH;

    sensorBaixoAtivo = digitalRead(sensorBaixo) == HIGH;

    // configura leds de acordo com o estado dos sensores

    digitalWrite(LED_1 , !sensorTopoAtivo && !sensorBaixoAtivo);

    digitalWrite(LED_2 , sensorBaixoAtivo && !sensorTopoAtivo);

    digitalWrite(LED_3 , sensorTopoAtivo && sensorBaixoAtivo);

    digitalWrite(LED_3 , sensorTopoAtivo);
```

```
delay(10);
```

```
// verifica se houve alteração no estado dos sensores
```

```
if (sensorTopoAtivo != ultimoSensorTopoAtivo || sensorBaixoAtivo !=  
ultimoSensorBaixoAtivo) {
```

```
    if (sensorTopoAtivo == HIGH) {
```

```
        Serial.println("Sensor topo ativo");
```

```
        //Tom simples para o buzzer
```

```
        tone(BUZZER, 1500, tempo);
```

```
        // chama função de inserção de infos no banco de dados
```

```
        insereInfosNoMySQL("Nível: Cheio");
```

```
        // chama funação para exibir no display que os dados foram salvos
```

```
        dados();
```

```
        // limpa display
```

```
        display.clear();
```

```
        display.setTextAlignment(TEXT_ALIGN_CENTER);
```

```
        // seleciona a fonte
```

```
        display.setFont(ArialMT_Plain_16);
```

```
        // seta no display a situação do reservatório
```

```
display.drawString(63, 10, "Nível de Agua:");
```

```
display.drawString(63, 26, "Cheio");
```

```
display.display();
```

```
}
```

```
else if (sensorTopoAtivo == LOW && sensorBaixoAtivo == HIGH) {
```

```
    Serial.println("Sensor baixo ativo");
```

```
    tone(BUZZER, 1300, tempo);
```

```
    insereInfosNoMySQL("Nível: Estável");
```

```
    dados();
```

```
    display.clear();
```

```
    display.setTextAlignment(TEXT_ALIGN_CENTER);
```

```
    display.setFont(ArialMT_Plain_16);
```

```
    display.drawString(63, 10, "Nível de Agua:");
```

```
    display.drawString(63, 26, "Estável");
```

```
    display.display();
```

```
}

else {

    Serial.println("Sensores inativos");

    tone(BUZZER, 1100, tempo);

    insereInfosNoMySQL("Nível: Crítico");

    dados();

    display.clear();

    display.setTextAlignment(TEXT_ALIGN_CENTER);

    display.setFont(ArialMT_Plain_16);

    display.drawString(63, 10, "Nível de Agua:");

    display.drawString(63, 26, "Crítico");

    display.display();

}

}
```

```
// variáveis para guardar última checagem dos sensores
```

```
ultimoSensorTopoAtivo = sensorTopoAtivo;
```

```
ultimoSensorBaixoAtivo = sensorBaixoAtivo;
```

```
delay(1000);
```

```
}
```

```
// função para realizar a inserção de dados ao banco de dados
```

```
void insereInfosNoMySQL(const char* situacao) {
```

```
    // atribui inserção ao banco de dados conectado
```

```
    MySQL_Query query_mem = MySQL_Query(&conn);
```

```
// variável para guardar o comando de inserção MySQL que será utilizado
```

```
    String INSERT_SQL = "INSERT INTO relatorio (data_hora, situacao) VALUES  
(DATE_SUB(NOW(), INTERVAL 3 HOUR), "" + String(situacao) + "");";
```

```
    Serial.println("Enviando informações para o banco de dados: ");
```

```
    Serial.println(INSERT_SQL);
```

```
// executa comando de inserção no mySQL e exibe a mensagem se deu certo ou errado
```



```
if (!query_mem.execute(INSERT_SQL.c_str())) {
```

```
    Serial.println("Erro de inserção.");
```

```
}
```

```
else {
```

```
    Serial.println("Inserção concluída.");
```

```
}
```

```
}
```

```
// função para exibir no display que os dados foram salvos
```

```
void dados() {
```

```
//Mensagem informativa na mudança de estado da boia
```

```
display.clear();
```

```
display.setTextAlignment(TEXT_ALIGN_CENTER);
```

```
//Seleciona a fonte
```

```
display.setFont(ArialMT_Plain_16);

display.drawString(63, 26, "Informação Salva!");

display.display();

delay (2000);

display.clear();

display.setTextAlignment(TEXT_ALIGN_CENTER);

display.setFont(ArialMT_Plain_16);

display.drawString(63, 15, "Enviado para");

display.drawString(63, 30, "Base de Dados");

display.display();

delay (2000);

}

:
```

## **APLICAÇÃO WEB**

Aplicação “Java Web”, nas linguagens HTML, CSS e Java, que se relacionará com o BD. As informações armazenadas serão exibidas na aplicação web desenvolvida. As seguintes imagens

representam o front-end.

Página inicial: apresenta o nome do site e o acesso às funcionalidades de visualização da última atualização, listagem do histórico e remoção de dados.

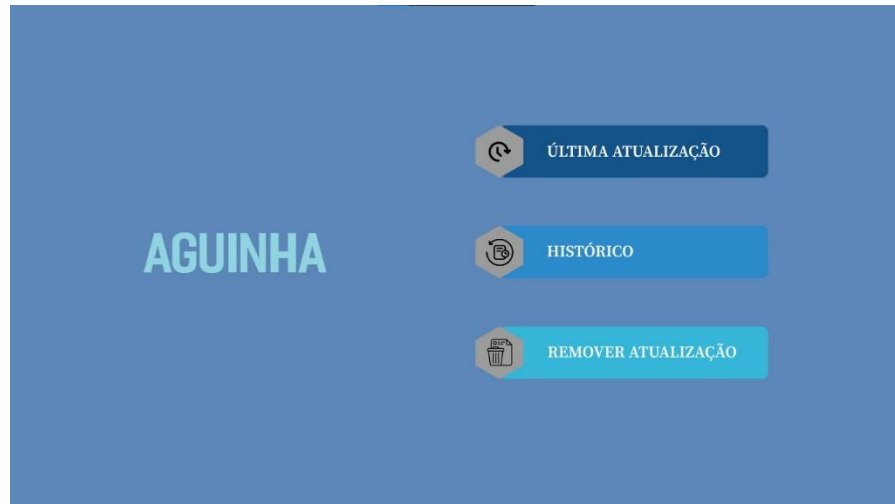


Figura 6: Home

Página de “Última Atualização”: Faz a exibição do registro mais recente inserido no banco de dados. A tabela exibe o ID, a data e horário, a situação do reservatório e a opção de excluir.



Figura 7: Tela "Última Atualização"

Página de “Histórico de Atualizações”: Exibe todos os dados registrados no banco de dados. A tabela exibe o ID, a data e horário, a situação do reservatório e a opção de excluir.



ID	Data	Hora	Situação	
10	27/11/2023	21:16:00	Crítico	<a href="#">Excluir</a>
9	27/11/2023	21:15:44	Estável	<a href="#">Excluir</a>
8	27/11/2023	21:14:31	cheio	<a href="#">Excluir</a>

Figura 8: Tela "Histórico de Atualizações"

Página de Remoção de Dados: Apresenta a opção de excluir o registro do banco de dados. O ID do registro será informado, e então, excluído. Se o ID não existir, a aplicação informa que o ID não foi encontrado.



Figura 9: Tela "Remoção de Dados"

## CÓDIGO DOS ARQUIVOS WEB

Todos os códigos estão disponíveis no GitHub. Acesso em: <https://github.com/JoaoF1610/PI-2-Nivel-de-Agua-.git>

Código em linguagem HTML. Constrói a página inicial do site. Arquivo “index.html” apresentado a seguir:

```
index.html X remove.html ultimaAtt.jsp remove.jsp historico.jsp
index.html > html
1 <html>
2   <head>
3     <title>Aguinha</title>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <link rel="stylesheet" href="estilos/indexStyle.css"/>
7   </head>
8   <body>
9     <main>
10      <h1 class="aguinha">AGUINHA</h1>
11
12      <a class="botao-1" href="ultimaAtt.jsp">
13        
14      </a>
15
16      <a href="historico.jsp" class="botao-2">
17        
18      </a>
19
20      <a class="botao-3" href="remove.html">
21        
22      </a>
23    </main>
24  </body>
25 </html>
26
27
```

Figura 10: Código HTML

Código em linguagem HTML. Constrói a página de Exclusão de Dados. A seguinte imagem mostra o arquivo “remove.html”:

```
index.html X remove.html X ultimaAtt.jsp remove.jsp historico.jsp
remove.html > html
1 <html>
2   <head>
3     <title>Remover</title>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <link rel="stylesheet" href="estilos/removeStyle.css"/>
7   </head>
8   <body>
9     <main>
10      <a href="http://localhost:8080/PI2-Aguinha/" class="aguinhalink">
11        <h1 class="aguinha">AGUINHA</h1>
12      </a>
13      <h2>Remoção de Dados</h2>
14      <form method="get" action="remove.jsp" class="formulario">
15        <label for="id_checagem">ID da checagem que deseja remover: </label>
16        <input type="number" name="id_checagem" id="id_checagem" required>
17        <input type="submit" name="remover">
18      </form>
19    </main>
20  </body>
21 </html>
22
```

Figura 11: Exclusão de Dados HTML

Arquivo JSP, para a opção de Exclusão de Dados:

```
index.html remove.html ultimaAtt.jsp remove.jsp x historico.jsp
remove.jsp > ?
1 <%@page import="java.sql.Connection"%>
2 <%@page import="java.sql.PreparedStatement"%>
3 <%@page import="java.sql.DriverManager"%>
4 <%@page contentType="text/html" pageEncoding="UTF-8"%>
5 <!DOCTYPE html>
6 <html>
7 <head>
8 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9 <title>Remover</title>
10 <link rel="stylesheet" href="estilos/removeStyle.css"/>
11 </head>
12 <body>
13 <main>
14 <%
15     int id_checagem;
16     id_checagem = Integer.parseInt(request.getParameter("id_checagem"));
17
18     try {
19         //Fazer a conexão com o Banco de Dados
20         Connection conecta;
21         PreparedStatement st;
22         Class.forName("com.mysql.cj.jdbc.Driver");
23         conecta = DriverManager.getConnection("jdbc:mysql://85.10.205.173:3306/reservatorio", "grupopi", "gjjlm123");
24
25         st = conecta.prepareStatement("DELETE FROM relatorio WHERE id_checagem = ?");
26         st.setInt(1, id_checagem);
27         int resultado = st.executeUpdate();
28     }
29 }
```

Figura 12: Exclusão de Dados JSP 1

```
28
29
30     if (resultado != 1) {
31         out.print("<a href='http://localhost:8080/PI2-Aguinha/'><h1 class='aguinha'>AGUINHA</h1></a>");
32         out.print("<p class='msg2'>Este ID não está cadastrado!</p>");
33     } else {
34         out.print("<a href='http://localhost:8080/PI2-Aguinha/'><h1 class='aguinha'>AGUINHA</h1></a>");
35         out.print("<p class='msg1'>O registro de ID " + id_checagem + " foi excluído com sucesso!</p>");
36     }
37 } catch (Exception x) {
38     out.print("Erro: " + x);
39 }
40 %>
41 </main>
42 </body>
43 </html>
44
```

Figura 13: Exclusão de Dados JSP 2

Arquivo JSP, para a exibição da última atualização do banco de dados:

```

1  <%@page import="java.sql.Connection"%>
2  <%@page import="java.sql.PreparedStatement"%>
3  <%@page import="java.sql.DriverManager"%>
4  <%@page import="java.sql.ResultSet"%>
5  <%@page contentType="text/html" pageEncoding="UTF-8"%>
6  <!DOCTYPE html>
7
8  <head>
9      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
10     <title>Última Atualização</title>
11     <link rel="stylesheet" href="estilos/ultimaAttStyle.css"/>
12 </head>
13 <body>
14     <main>
15         <a href="http://localhost:8080/PI2-Aguinha/" class="aguinhalink">
16             <h1 class="aguinha">AGUINHA</h1>
17         </a>
18         <h2>Última Atualização</h2>
19         <%
20             try {
21                 //Fazer a conexão com o Banco de Dados
22                 Connection conecta;
23                 PreparedStatement st;
24                 Class.forName("com.mysql.cj.jdbc.Driver");
25                 conecta = DriverManager.getConnection("jdbc:mysql://85.10.205.173:3306/reservatorio", "grupopi", "gjjlm123");
26
27                 st = conecta.prepareStatement("SELECT id_cheragem, DATE_FORMAT(data_hora, '%d/%m/%Y') AS Data, TIME_FORMAT(data_hora, '%H:%i:%s') AS Hora, situa
28                 ResultSet rs = st.executeQuery();
29             }

```

Figura 14: Última Atualização JSP 1

```

29     <%>
30     <table>
31         <tr>
32             <strong> <th>ID</th> <th>Data</th> <th>Hora</th> <th>Situação</th> </strong>
33         </tr>
34         <%
35             while (rs.next()) {
36                 <%
37                     <tr>
38                         <td><%= rs.getString("id_cheragem")%></td>
39                         <td><%= rs.getString("Data")%></td>
40                         <td><%= rs.getString("Hora")%></td>
41                         <td><%= rs.getString("Situação")%></td>
42                         <td><a style="text-decoration: none; color: blue;"
43                             onmouseover="this.style.color = 'red';"
44                             onmouseout="this.style.color = 'blue';"
45                             href="remove.jsp?id_cheragem=<%= rs.getString("id_cheragem")%>">Excluir</a></td>
46                     </tr>
47                 <%
48             }
49         <%>
50     </table>
51     <%
52     } catch (Exception x) {
53         out.print("Erro:" + x.getMessage());
54     }
55     <%>
56 </main>
57 </body>

```

Figura 15: Última Atualização JSP 2

Arquivo JSP para visualização do histórico de dados incluídos no banco de dados:

```

1  <%@page import="java.sql.Connection"%>
2  <%@page import="java.sql.PreparedStatement"%>
3  <%@page import="java.sql.DriverManager"%>
4  <%@page import="java.sql.ResultSet"%>
5  <%@page contentType="text/html" pageEncoding="UTF-8"%>
6  <!DOCTYPE html>
7  <html>
8  <head>
9      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
10     <title>Histórico</title>
11     <link rel="stylesheet" href="estilos/ultimaAttStyle.css"/>
12 </head>
13 <body>
14     <main>
15         <a href="http://localhost:8080/PI2-Aguinha/" class="aguinhalink">
16             <h1 class="aguinha">AGUINHA</h1>
17         </a>
18         <h2>Histórico de Atualizações</h2>
19         <%
20             try {
21                 //Fazer a conexão com o Banco de Dados
22                 Connection conecta;
23                 PreparedStatement st;
24                 Class.forName("com.mysql.cj.jdbc.Driver");
25                 conecta = DriverManager.getConnection("jdbc:mysql://85.10.205.173:3306/reservatorio", "grupopi", "gjjlm123");
26
27                 st = conecta.prepareStatement("SELECT id_checagem, DATE_FORMAT(data_hora, '%d/%m/%Y') AS Data, TIME_FORMAT(data_hora, '%H:%i:%s') AS Hora, situa
28                 ResultSet rs = st.executeQuery();
29             }

```

Figura 16: Histórico de Dados JSP 1

```

30     <table>
31         <tr>
32             <strong> <th>ID</th> <th>Data</th> <th>Hora</th> <th>Situação</th> </strong>
33         </tr>
34         <%
35             while (rs.next()) {
36                 <%
37                     <tr>
38                         <td><%= rs.getString("id_checagem")%></td>
39                         <td><%= rs.getString("Data")%></td>
40                         <td><%= rs.getString("Hora")%></td>
41                         <td><%= rs.getString("Situação")%></td>
42                         <td><a style="text-decoration: none; color: blue;"
43                             onmouseover="this.style.color = 'red';"
44                             onmouseout="this.style.color = 'blue';"
45                             href="remove.jsp?id_checagem=<%= rs.getString("id_checagem")%>">Excluir</a></td>
46                     </tr>
47                 <%
48                     }
49                 <%
50             </table>
51             <%
52             } catch (Exception x) {
53                 out.print("Erro:" + x.getMessage());
54             }
55             <%
56         </main>
57     </body>
58 </html>

```

Figura 17: Histórico de Dados JSP 2



## CÓDIGO CSS

Código em linguagem CSS referente ao arquivo “index.html”.

```
# indexStyle.css X # removeStyle.css # ultimaAttStyle.css
web > estilos > # indexStyle.css > ...
1  @import url('https://fonts.cdnfonts.com/css/fester');
2
3  * {
4      margin: 0;
5  }
6
7  main {
8      background-color: #5d87b9;
9      width: 100%;
10     height: 100%;
11 }
12
13 .aguinha {
14     font-size: 500%;
15     font-family: 'Fester', sans-serif;
16     color: #90d5e1;
17     position: relative;
18     top: 48%;
19     transform: translateY(-50%);
20     margin-left: 15%;
21 }
22
23 .botao-1,
24 .botao-2,
25 .botao-3 {
26     border: none;
27     position: absolute;
28     right: 15%;
29     transition: background-color 0.5s ease, color 0.5s ease, transform 0.5s ease;
30 }
31
32 .botao-1 {
33     top: 24%;
34 }
35
```

Figura 18: CSS index

```
# indexStyle.css X # removeStyle.css # ultimaAttStyle.css
web > estilos > # indexStyle.css > .botao-1:~hover
34 }
35
36 .botao-2 {
37     top: 44%;
38 }
39
40 .botao-3 {
41     top: 64%;
42 }
43
44 .botao-1 .img,
45 .botao-2 .img,
46 .botao-3 .img {
47     width: 450px;
48     height: 80px;
49     transition: transform 0.5s ease;
50 }
51
52 .botao-1:~hover,
53 .botao-2:~hover,
54 .botao-3:~hover {
55     transform: scale(1.1);
56 }
57
58 .botao-1:~hover .img,
59 .botao-2:~hover .img,
60 .botao-3:~hover .img {
61     transform: scale(1.1);
62 }
```

Figura 19: CSS index continuação

Código referente ao arquivo “remove.html”



The image shows a code editor with three tabs: # indexStyle.css, # removeStyle.css (active), and # ultimaAttStyle.css. The active tab displays CSS code for 'removeStyle.css'. The code includes an @import statement for a font, a reset for all elements, and styles for 'main', 'body', '.aguinha', and '.aguinhalink'.

```
web > estilos > # removeStyle.css > ...
1  @import url('https://fonts.cdnfonts.com/css/fester');
2
3  * {
4      margin: 0;
5  }
6
7  main {
8      text-align: center;
9      background-color: #5d87b9;
10     width: 100%;
11     height: 100%;
12 }
13
14 body {
15     text-align: center;
16     background-color: #5d87b9;
17     width: 100%;
18     height: 100%;
19 }
20
21 .aguinha {
22     font-size: 500%;
23     font-family: 'Fester', sans-serif;
24     color: #90d5e1;
25     position: relative;
26     display: inline-block;
27 }
28
29 .aguinhalink {
30     text-decoration: none;
31 }
32
```

Figura 20: CSS remove.html



```
32
33 h2 {
34     margin-top: 2%;
35 }
36
37 .formulario {
38     margin-top: 2%;
39 }
40
41 .msg1 {
42     color:  green;
43     margin-top: 2%;
44 }
45
46 .msg2 {
47     color:  red;
48     margin-top: 2%;
49 }
```

Figura 21: CSS remove.html continuação

Código referente ao arquivo “ultimaAtt.jsp” e “historico.jsp”:



```
# indexStyle.css # removeStyle.css # ultimaAttStyle.css X
web > estilos > # ultimaAttStyle.css > ...
1  @import url('https://fonts.cdnfonts.com/css/fester');
2
3  * {
4      margin: 0;
5  }
6
7  body {
8      display: flex;
9      width: 100%;
10     height: 100%;
11     align-items: center;
12     justify-content: center;
13     background-color: #5d87b9;
14 }
15
16 main {
17     text-align: center;
18 }
19
20 .aguinha {
21     font-size: 500%;
22     font-family: 'Fester', sans-serif;
23     color: #90d5e1;
24     display: inline-block;
25 }
26
27 .aguinhalink {
28     text-decoration: none;
29 }
30
31 h2 {
32     margin-top: 6%;
33 }
34
```

Figura 22: CSS ultimaAtt.jsp e historico.jsp

```
30
31 h2 {
32 |   margin-top: 6%;
33 | }
34
35 table {
36 |   border-collapse: collapse;
37 |   width: 150%;
38 |   margin: 20px;
39 |   margin-left: -25%;
40 | }
41
42 th, td {
43 |   border: 1px solid #ddd;
44 |   padding: 8px;
45 | }
46
47 th {
48 |   background-color: #f2f2f2;
49 | }
50
51 td {
52 |   color: white;
53 | }
```

Figura 23: CSS ultimaAtt.jsp e historico.jsp

## **CONCLUSÃO**

Este trabalho apresentou o desenvolvimento do sistema a partir do planejamento feito nas datas anteriores. Conseguimos concluir todos os objetivos definidos nos trabalhos anteriores. Todos os elementos que compõem este trabalho foram concluídos com êxito. Estamos muito satisfeitos com os resultados da nossa dedicação e compromisso em atender todas as especificações do projeto. Incluímos os conhecimentos de todas as áreas estudadas desde o início do curso, até o momento, o que leva ao aperfeiçoamento das nossas habilidades e aprendizados. Conseguimos também, aperfeiçoar o projeto, adicionando elementos extras.

## REFERÊNCIAS

DAVIS, Stephen. C++ para Leigos. 7ª Edição. Editora Alta Books. Publicado em: 10 de junho de 2016.

OLIVEIRA, C; ZANEETI, H; NABARRO, C; GONÇALVES, J. Aprenda Arduino: Uma abordagem prática. 1ª Edição. Katzen Editora. Publicado em 2018.

RANGEL, Gabriel. Ebook Internet das Coisas para iniciantes. Eletrônica Ômega. Disponível em: <https://blog.arduinoomega.com/ebooks/Eletronica-Omega-Ebook-IoT-Para-Iniciantes.pdf>

Banco de Dados. Disponível em: <https://www.db4free.net/index.php?language=pt>