

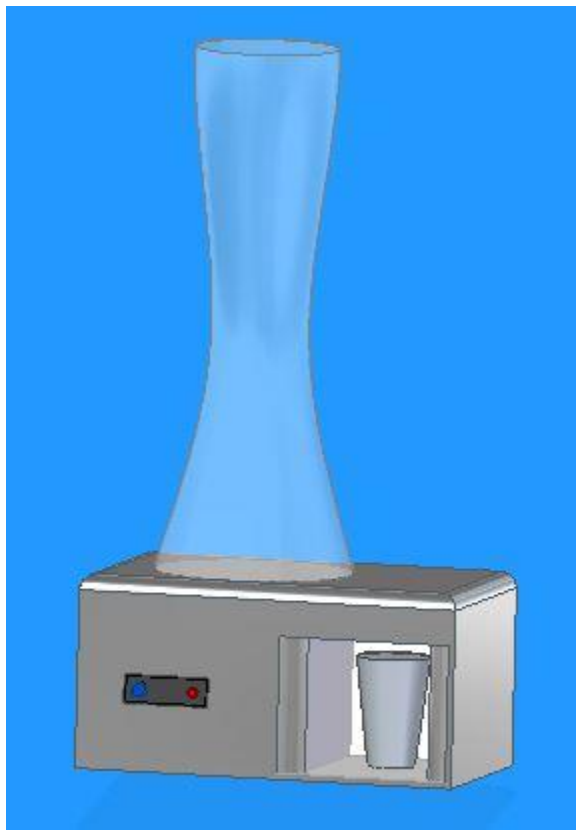
SISTEMA DE DOSAGEM COM O ESP8266:

Uma aplicação em dosagem automática de Chopp – BeerTower Chopp



**INSTITUTO
FEDERAL**
Paraíba

VISÃO DA IDEIA



1

INTRODUÇÃO E OBJETIVOS



INTRODUÇÃO E OBJETIVOS

- Este protótipo é um primeiro fruto de um projeto de um sistema de dosagem automática a ser implementado em torres de Chopp
- O usuário através de um push button pode dosar a quantidade de chopp de sua preferência em seu copo



INTRODUÇÃO E OBJETIVOS

- O estabelecimento terá acesso ao controle das chopeiras através de uma aplicação em QT em comunicação serial com o NODE MCU 8266.
- Já o cliente por sua vez, terá acesso a temperatura do Chopp, bem como a solicitação do garçom junto a mesa e o valor gasto através de uma página WEB que poderá ser acessada com qualquer dispositivo móvel com acesso a internet através de um QR Code.

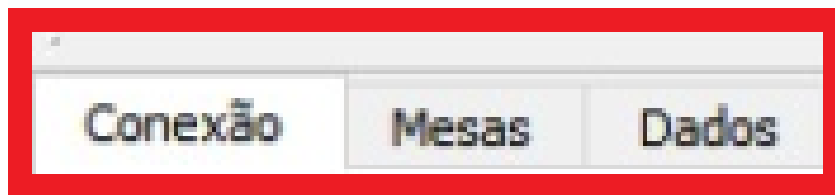
2

INTERFACE DO PROPRIETÁRIO



Interface do Proprietário (QT)

- A Interface do proprietário do estabelecimento é composto de três abas conforme mostra a figura a seguir:





Interface do Proprietário (QT)

■ Estabelecendo a conexão

Conexão
Mesas
Dados

Serial:

Porta
COM7

Velocidade:
115200

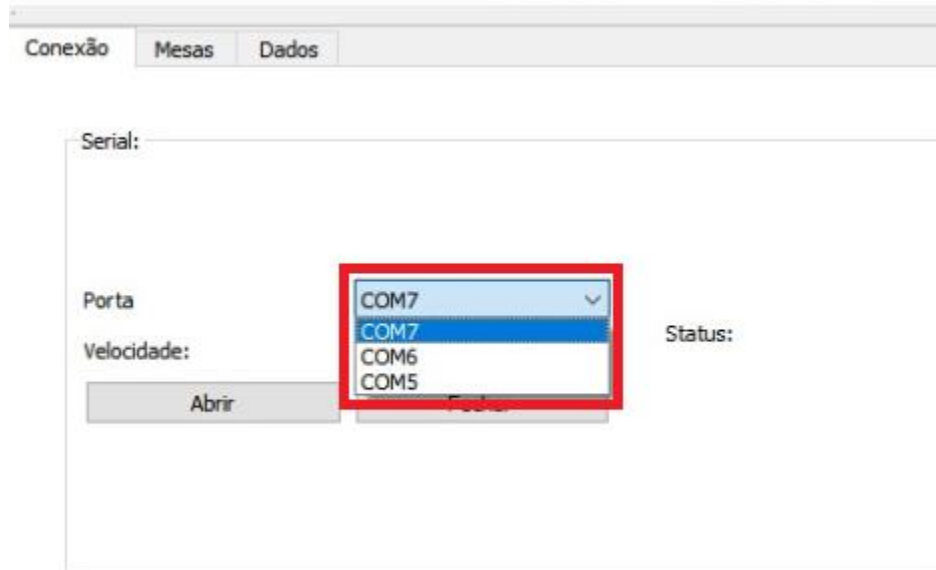
Abrir
Fechar

Status: Desconecta



Interface do Proprietário (QT)

■ Estabelecendo a conexão



Conexão Mesas Dados

Serial:

Porta

Velocidade:

Abrir

Status:

COM7
COM7
COM6
COM5



Interface do Proprietário (QT)

■ Estabelecendo a conexão

Conexão Mesas Dados

Serial:

Porta COM7

Velocidade:

Abrir

Status:

110

4800

9600

14400

19200

38400

56000

57600

115200

128000

256000



Interface do Proprietário (QT)

■ Estabelecendo a conexão

Conexão
Mesas
Dados

Serial:

Porta
COM7

Velocidade:
115200

Abrir
Fechar

Status: Desconectz



Interface do Proprietário (QT)

■ Estabelecendo a conexão

Conexão
Mesas
Dados

Serial:

Porta
COM7

Velocidade:
115200

Abrir
Fechar

Status: Conectado



Interface do Proprietário (QT)

Estabelecendo a Mesa do Cliente

Conexão
Mesas
Dados

MESA: 0

Mesa 0

Mesa 5

Mesa 1

Mesa 6

Mesa 2

Mesa 7

Mesa 3

Mesa 8

Mesa 4

Mesa 9

OCUPAR
DESOCUPAR

Pedido do cliente feito na chopeira: 1

SALVAR




Interface do Proprietário (QT)

Estabelecendo a Mesa do Cliente

Conexão Mesas Dados

MESA: 0 ▾
0
1
2
3
4
5
6
7
8
9



Mesa 0 Mesa 5

Mesa 1 Mesa 6

Mesa 2 Mesa 7

Mesa 3 Mesa 8

Mesa 4 Mesa 9

OCUPAR DESOCUPAR

Pedido do cliente feito na chopeira: 1 ▾ SALVAR




Interface do Proprietário (QT)

Estabelecendo a Mesa do Cliente





Conexão Mesas Dados

MESA: 4



OCUPAR DESOCUPAR

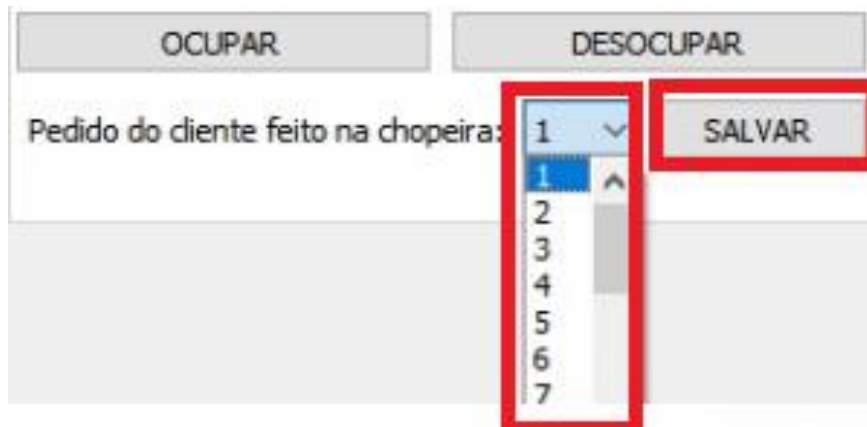
Pedido do cliente feito na chopeira: 1 SALVAR

Mesa 0	OCUPADO	Mesa 5	OCUPADO
Mesa 1	OCUPADO	Mesa 6	
Mesa 2	OCUPADO	Mesa 7	OCUPADO
Mesa 3		Mesa 8	
Mesa 4		Mesa 9	OCUPADO



Interface do Proprietário (QT)

■ Estabelecendo a Mesa do Cliente



OCUPAR DESOCUPAR

Pedido do cliente feito na chopeira:

1 2 3 4 5 6 7

SALVAR



Interface do Proprietário (QT)

Verificando os dados da Mesa / Choqueira

Conexão
Mesas
Dados

Dados dos clientes

	Quantidade	Valor
Mesa 0		
Mesa 1		
Mesa 2		
Mesa 3		
Mesa 4		
Mesa 5		

Chamados

Mesa

Histórico

Mesa	Valor	Client
------	-------	--------

< >

Pedido atendido

< >



Interface do Proprietário (QT)

Verificando os dados da Mesa / Chopeira

Dados dos clientes

	Quantidade	Valor
Mesa 0	1	49.9
Mesa 1		
Mesa 2		
Mesa 3		
Mesa 4		
Mesa 5		



Interface do Proprietário (QT)

Verificando os dados da Mesa / Chopeira : Chamado

Chamados

Mesa	
1	MESA 0

Pedido atendido



Interface do Proprietário (QT)

- Verificando os dados da Mesa / Chopeira : Chamado

Chamados

Mesa	
1	MESA 0

Pedido atendido

3

INTERFACE DO CLIENTE



Interface do Usuário (Botões físicos e Web)

- A interface física está localizada na própria chopeira que contem:
- Um botão para dosagem do chopp
- Um botão para solicitação ao estabelecimento
- Um LED para informar que uma solicitação foi realizada



Interface do Usuário (Botões físicos e Web)

■ A interface física





Interface do Usuário (Botões físicos e Web)

■ Interface Web

<http://beertowerchopp.herokuapp.com/>





Interface do Usuário (Botões físicos e Web)

Interface Web





Interface do Usuário (Botões físicos e Web)

■ Interface Web – Card Temperatura

Temperatura °C
27.94



Interface do Usuário (Botões físicos e Web)

Interface Web – Card Atendimento

Atendimento
<p>Chamar garçom</p> <div>ChamarCancelar</div>



Interface do Usuário (Botões físicos e Web)

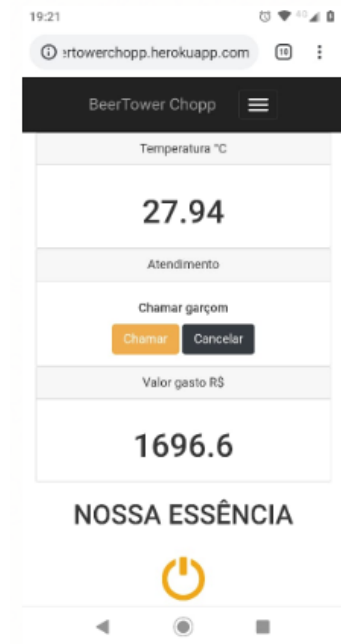
■ Interface Web – Valor Gasto

Valor gasto R\$
1696.6



Interface do Usuário (Botões físicos e Web)

Interface Web – Versão Mobile



4

Do Outro Lado do Protótipo



Do outro Lado do Protótipo

O desenvolvimento lógico do protótipo BeerTower-Chopp contém três segmentos básicos:

- O NodeMCU 8266: Que fará o controle dos dispositivos eletrônicos. Na figura abaixo podemos ver a conexão destes dispositivos utilizados no protótipo.
- A Interface do Estabelecimento: Desenvolvido através do QT Creator.
- A Interface do Cliente: Página WEB que poderá ser acessada pelo usuário.



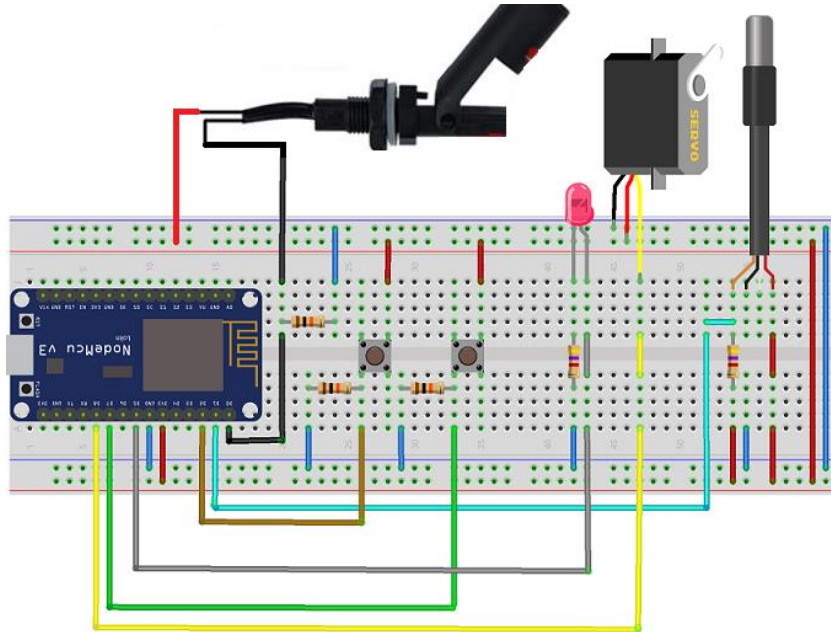
INTERFACE GRÁFICA – QT CREATOR

- Dispositivos Utilizados
- ✓ ESP 8266
- ✓ Sensor de Nível
- ✓ Sensor de Temperatura
- ✓ Botão
- ✓ LED
- ✓ Resistores
- ✓ Válvula com ServoMotor



Do outro Lado do Protótipo

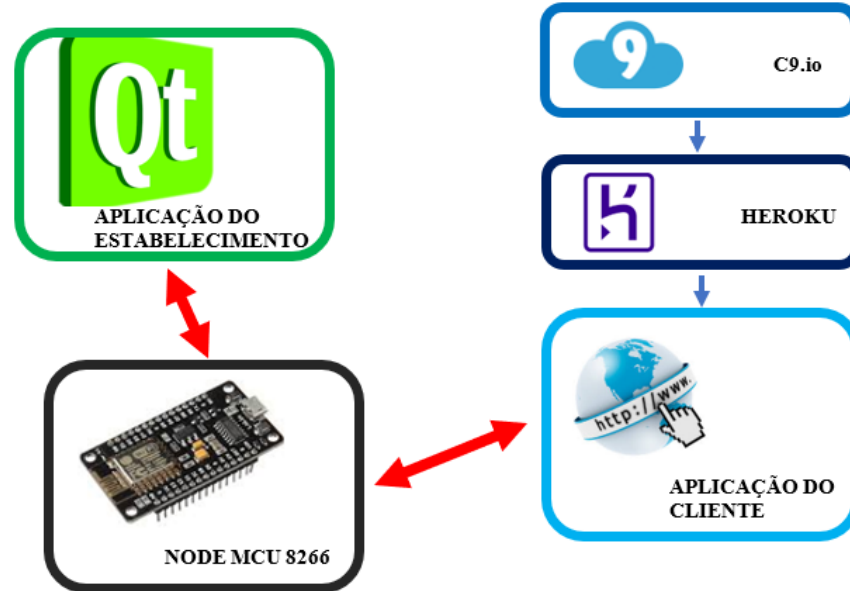
Esquema Eletrônico





Do outro Lado do Protótipo

Comunicação





Do outro Lado do Protótipo

■ Comunicação

A comunicação entre os diversos componentes do projeto está baseada no protocolo Json.

JSON (JavaScript Object Notation)



Do outro Lado do Protótipo

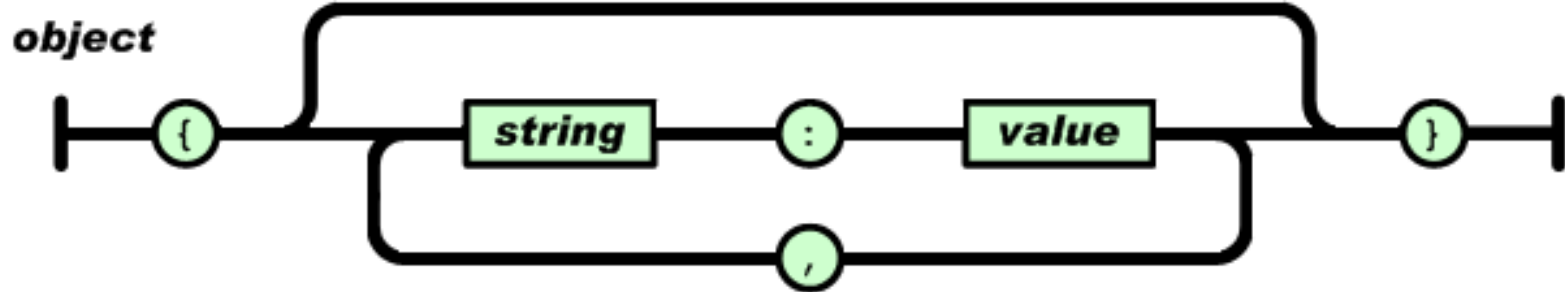
■ Comunicação

No JSON, um objeto é um conjunto não ordenado de pares nome / valor. Um objeto começa com { (chave esquerda) e termina com } (chave direita) . Cada nome é seguido por : e o nome / pares de valores estão separados por , (vírgula) .



Do outro Lado do Protótipo

Comunicação - Json





Do outro Lado do Protótipo

Comunicação – Json

```
var socket = io.connect('http://beertowerchopp.herokuapp.com/');
socket.on("atualiza", function(dados) {
  console.log(dados);
  if("TEMPERATURA" in dados){
    console.log("temperatura");
    document.getElementById("temperatura").innerHTML = dados["TEMPERATURA"]

  }else if("VALOR" in dados){
    console.log("valor_cliente");
    document.getElementById("valor_cliente").innerHTML = dados["VALOR"]

  } else {

    console.log("temperatura");
    document.getElementById("temperatura").innerHTML = dados["TEMPERATURA"]
  }
});
```

Atualização de
Temperatura e
Valor no Site



Do outro Lado do Protótipo

Comunicação – Json

```
estado = {
    "TEMPERATURA" : 0,
    "PAGAMENTO" : 0,
    "VALOR" : 0 ,
    "CHAMADO" : 0
}

change = 0;
```

O Json é definido no app.py com os valores zerados e Change também vai ser inicialmente zero.



Do outro Lado do Protótipo

Comunicação – Json

```
@app.route("/upload", methods=["POST"])
def rota_data():
    global estado
    estado = request.get_json()
    socketio.emit("atualiza",estado)
    return "ok"

@app.route("/download", methods=["GET"])
def rota_download():
    global estado
    global change
    if change == 1 :
        change = 0
        return json.dumps(estado)
    return "1";
```

No app.py também é definido as rotas de download e upload



Do outro Lado do Protótipo

Comunicação – Json

```
@app.route("/upload", methods=["POST"])
def rota_data():
    global estado
    estado = request.get_json()
    socketio.emit("atualiza", estado)
    return "ok"

@app.route("/download", methods=["GET"])
def rota_download():
    global estado
    global change
    if change == 1 :
        change = 0
        return json.dumps(estado)
    return "1";
```

Download, é voltado para o recebimento de dados do servidor por meio de Post, formados por Json's, e o Upload, versa para a atualização de dados dentro do servidor.



Do outro Lado do Protótipo

Comunicação – Json

```
@socketio.on('chamar_garçom')
def Chamar_garçom():
    global estado
    global change
    change = 1
    estado["CHAMADO"] = 1

@socketio.on('cancelar_garçom')
def Cancelar_chamada():
    global estado, change
    change = 1;
    estado["CHAMADO"] = 0
```

socketio.on vai mandar um json a partir de alguma alteração no estado dos buttons no site



Do outro Lado do Protótipo

Código – NODEMECU 8266

```
#include <ESP8266HTTPClient.h>
#include <ESP8266WiFi.h>
const char* SSID = "JSNFNDB";
const char* PASS = "t06iohph";
HTTPClient http;
const String LOCATION = "http://beertowerchopp.herokuapp.com/";
const String UPLOAD = "upload";
const String DOWNLOAD = "download";

const String TEMPERATURA = "\"TEMPERATURA\":";
const String CHAMADO = "\"CHAMADO\":";
const String VALOR = "\"VALOR\":";

#include <DallasTemperature.h>
#include <OneWire.h>
#define ONE_WIRE_BUS 5 //D1 pin of nodemcu
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
```

Características da Rede
WiFi e das Bibliotecas
Importadas.



Do outro Lado do Protótipo

■ Código – NODEMECU 8266

```
String downloadWEB() {  
    http.begin(LOCATION+DOWNLOAD) ;  
    http.GET() ;  
    String payload = http.getString() ;  
    http.end() ;  
    return payload;  
}
```



Do outro Lado do Protótipo

■ Código – NODEMECU 8266

```
my_json = downloadWEB();
if(my_json!="1") {
  Serial.print(my_json);
  if(my_json.indexOf("\"CHAMADO\": 1") != -1) {
    digitalWrite(led_1, HIGH);
  } else if(my_json.indexOf("\"CHAMADO\": 0") != -1) {
    digitalWrite(led_1, LOW);
  }
}
```



Do outro Lado do Protótipo

Código – NODEMECU 8266

```

if(digitalRead(sensor_de_nivel) == HIGH){
  digitalWrite(led_1, HIGH);
  Serial.print(JSON_CHAMADO());
  delay(1000);

}else if(digitalRead(btn_1)==HIGH){  // botão para chamado
  digitalWrite(led_1, HIGH);
  Serial.print(JSON_CHAMADO());
}else{

uploadTEMPERATURA();
delay(50);
uploadVALOR();
delay(50);

}

```

Ativando o chamado



Do outro Lado do Protótipo

■ Código – NODEMECU 8266

```
String JSON_CHAMADO() {
    return "{" +
    CHAMADO + "1" +
    "}";
}
```

Criação do Json
Chamado



Do outro Lado do Protótipo

■ Código – NODEMECU 8266

```
String JSON_TEMP() {
  sensors.requestTemperatures();
  float temp = sensors.getTempCByIndex(0);
  return "{" +
  TEMPERATURA + String(temp) +
  "}";
}
```

Criação do Json
Temperatura



Do outro Lado do Protótipo

■ Código – NODEMECU 8266

```
void uploadTEMPERATURA() {
  http.begin(LOCATION+UPLOAD);
  http.addHeader("Content-Type", "application/json");
  http.POST(JSON_TEMP());
  Serial.print(JSON_TEMP());
  http.end();
}
```

```
void uploadCHAMADO() {
  http.begin(LOCATION+UPLOAD);
  http.addHeader("Content-Type", "application/json");
  http.POST(JSON_CHAMADO());
  Serial.print(JSON_CHAMADO());
  http.end();
}
```

Rotas upload e Download

OBRIGADO PELA ATENÇÃO!

APOLO DE LIMA (20181610009)
EDER MADRUGA COELHO (20181610028)
EVERTON JÚNIOR DA SILVA ARRUDA (20181610044)
IVIS FERREIRA DE BRITO (20181610013)