Escola Secundaria Manuel Teixeira Gomes

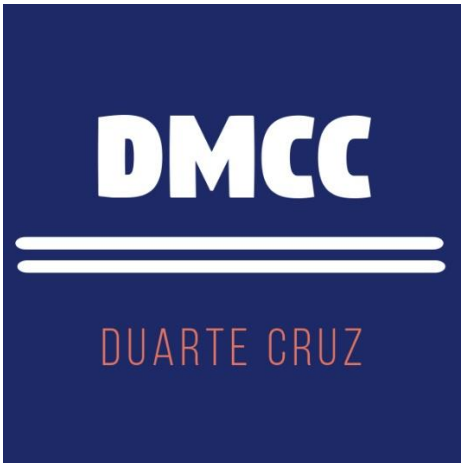
Técnico de gestão e programação de sistemas informáticos

2017/2020

WebSockets: Gestão de filas “Hospital MagoPT”

Um projeto de

Duarte Manuel Cabrita da Cruz



No âmbito da disciplina de redes da comunicação

Março de 2020

Contacto:

Duarte Manuel Cabrita da Cruz

Escola Secundaria Manuel Teixeira Gomes

Email: a28254@aemtg.pt

“WebSockets: Gestão de filas “Hospital MagoPT”

Índice

Conteúdo

Índice de imagens

Introdução

Este projeto foi desenvolvido no âmbito da disciplina de redes da comunicação com o objetivo de explorar as possibilidades dos WebSockets.

Software Utilizado

Para o desenvolvimento deste projeto foram utilizados os programas:

- Sistemas Operativos:
  - Ubuntu 20.04
    - Usado para hospedar o servidor
    - Utilizado para o desenvolvimento do código
  - Windows 10 Pro
    - Utilizado para o desenvolvimento de código
- IDE's
  - Pycharm
  - Intelij Idea
- Browser's
  - Firefox
  - Microsoft Edge
  - Google Chrome

Linguagens e tecnologias

As linguagens utilizadas para este projeto são as seguintes:

- Server Side
  - Python 3.7
  - PHP
  - JSON
- Client Side
  - HTML

- CSS
- JavaScript

- Framework
  - Bootstrap

- Serviços e Protocolos utilizados
  - Web: Apache2
  - File Transfer: FTP
  - Conexão remota: SSH
  - Gestão Data base: phpmyadmin
  - Data base: Mysql

Preparar o Python

Para se trabalhar em WebSockets no python existem duas bibliotecas essenciais sendo elas instaladas através dos seguintes comandos

```
pip install asyncio #Biblioteca que vai permitir a sincronia na aplicação
pip install websockets #Biblioteca que vai permitir o uso de WebSockets
```

Data base

A Data base utilizada neste projeto é bastante simples sendo constituída apenas de uma tabela onde será guardada as informações dos médicos, essa DB pode ser criada usando o seguinte código:

```
CREATE TABLE `medicos` (
    `id_med` int NOT NULL,
    `nome` varchar(90) NOT NULL,
    `apelido` varchar(90) NOT NULL,
    `pass` varchar(350) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT
NULL,
    `email` varchar(250) DEFAULT NULL,
    `especializacao` varchar(90) NOT NULL,
    `registo` varchar(90) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Código

Back end

Código de terceiros

Algum do código python utilizado neste projeto foi retirado do da internet sendo esse código o seguinte:

```
def state_event():
    return json.dumps({"type": "state", **STATE})


def users_event():
    return json.dumps({"type": "users", "count": len(USERS)})


async def notify_state():
    if USERS:  # asyncio.wait doesn't accept an empty list
        message = state_event()
        await asyncio.wait([user.send(message) for user in USERS])


async def notify_users():
    if USERS:  # asyncio.wait doesn't accept an empty list
        message = users_event()
        await asyncio.wait([user.send(message) for user in USERS])


async def register(websocket):
    USERS.add(websocket)
    await notify_users()


async def unregister(websocket):
    USERS.remove(websocket)
    await notify_users()
```

Este Código serve como base para trabalhar em WebSockets no python sendo que este código trata do registo de novos clientes(máquinas) e atualiza-os.

O código original pode ser encontrado em: https://websockets.readthedocs.io/en/stable/intro.html

Código python

O código criado propositadamente para este projeto foi o seguinte:

Bibliotecas utilizadas:

```
import asyncio
import json
import logging
import websockets
```

Para a criação de uma variável de armazenamento:

```
ip = "192.168.1.85"
ordem = []
STATE = {"total_geral": 0, "atual_geral":0,"total_oftalmologista": 0,
"atual_oftalmologista":0,"total_cardiologia": 0,
"atual_cardiologia":0,"total_psicologia": 0,
"atual_psicologia":0,"status":'done',"som":"","last":"","ordem":ordem}
```

Atualização após ordem do utilizador:

```
async def counter(websocket, path):
    # register(websocket) sends user_event() to websocket
    await register(websocket)
    try:
        await websocket.send(state_event())
```

```
        async for message in websocket:
            data = json.loads(message)
            #Registo do medico
            print(message)
            if data["action"] == "senha":
                if data["esp"] == "ger":
                    STATE['total_geral'] += 1
                    STATE['last'] = "ger"

                elif data["esp"] == "oft":
                    STATE['total_oftalmologista'] += 1
                    STATE['last'] = "oft"

                elif data["esp"] == "car":
                    STATE['total_cardiologia'] += 1
                    STATE['last'] = "car"

                elif data["esp"] == "psi":
                    STATE['total_psicologia'] += 1
                    STATE['last'] = "psi"

                STATE["status"] = 'done'
                STATE["som"] = ''
                await notify_state()

            elif data["action"] == "proximo":

                if data["esp"] == "Geral":
                    if STATE['atual_geral'] < STATE['total_geral']:
                        STATE['atual_geral'] += 1
                        STATE["status"] = 'done'
                        STATE['last'] = ""
                        STATE["som"] = "sound.mp3"
                        info = {"nome" :
data["medico"],"senha":STATE["atual_geral"],"especialidade":"geral"}
                        ordem.append(info)
                        STATE['ordem']=ordem[-5:]
                    else:
                        STATE["som"] = ""
                        STATE["status"] = 'error'
                    await notify_state()

                elif data["esp"] == "Oftalmologista":
                    if STATE['atual_oftalmologista'] <
STATE['total_oftalmologista']:
                        STATE['atual_oftalmologista'] += 1
                        STATE["status"] = 'done'
                        STATE['last'] = ""
                        STATE["som"] = "sound.mp3"
                        info = {"nome" :
data["medico"],"senha":STATE["atual_oftalmologista"],"especialidade":"Oftalmo
logista"}
                        ordem.append(info)
                        STATE['ordem']=ordem[-5:]
                    else:
                        STATE["som"] = ""
                        STATE["status"] = 'error'
                    await notify_state()
```

```
                elif data["esp"] == "Cardiologia":
                    if STATE['atual_cardiologia'] <
STATE['total_cardiologia']:
                        STATE['atual_cardiologia'] += 1
                        STATE["status"] = 'done'
                        STATE['last'] = ""
                        STATE["som"] = "sound.mp3"
                        info = {"nome" :
data["medico"],"senha":STATE["atual_cardiologia"],"especialidade":"Cardiologi
a"}
                        ordem.append(info)
                        STATE['ordem']=ordem[-5:]
                    else:
                        STATE["som"] = ""
                        STATE["status"] = 'error'
                    await notify_state()

                elif data["esp"] == "Psicologia":
                    if STATE['atual_psicologia'] < STATE['total_psicologia']:
                        STATE['atual_psicologia'] += 1
                        STATE['last'] = ""
                        STATE["status"] = 'done'
                        STATE["som"] = "sound.mp3"
                        info = {"nome" :
data["medico"],"senha":STATE["atual_psicologia"],"especialidade":"Psicologia"
}
                        ordem.append(info)
                        STATE['ordem']=ordem[-5:]
                    else:
                        STATE["som"] = ""
                        STATE["status"] = 'error'
                    await notify_state()


            else:
                logging.error("unsupported event: {}", data)
    finally:
        await unregister(websocket)
```

Este é todo o código “backbone” do projeto

Front end

Conexão a Data Base

```
<?php
$db_host = "localhost";
$db_name = ""; //nome da DB
$db_user = ""; //username da DB
$db_pass = ""; //Password do user a DB

try{
    $db_con = new
PDO("mysql:host={$db_host};dbname={$db_name}",$db_user,$db_pass);
    $db_con->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
}catch(PDOException $e){
    echo $e->getMessage();
}
?>
```

Funções Java Script

Declaração de variáveis

```
var ip = "192.168.1.85"; //ip do servidor
var porta = "6788"; //Porta usada pelo protocolo
var script = document.createElement('script');
var som; //Variável
script.src = 'https://code.jquery.com/jquery-3.4.1.min.js';
script.type = 'text/javascript';
document.getElementsByTagName('head')[0].appendChild(script)
```

Função som

Esta função é usada para reproduzir um som sempre que uma nova senha é chamada

```
function sound(src) {
    this.sound = document.createElement("audio");
    this.sound.src = src;
    this.sound.setAttribute("preload", "auto");
    this.sound.setAttribute("controls", "none");
    this.sound.style.display = "none";
    document.body.appendChild(this.sound);
    this.play = function () {
        this.sound.play();
    }
    this.stop = function () {
        this.sound.pause();
    }
}
```

Funções usadas no relógio

O relógio utiliza duas funções diferentes para funcionar sendo elas:

```
function checkTime(i) {
    if (i < 10) {
        i = "0" + i
    }
    ;  // adicona um zero a frente de numeros < 10
    return i;
};

function relogio() {
    var today = new Date();
    var h = today.getHours();
    var m = today.getMinutes();
    var s = today.getSeconds();
    m = checkTime(m);
    s = checkTime(s);
    document.getElementById('relogio').innerHTML =
        h + ":" + m + ":" + s;
    var t = setTimeout(relogio, 500);
}
```

Index

O index da aplicação utiliza 2 funções para se atualizar sendo uma delas usada apenas para desmembrar um array:

```
function print(data) {
    var a = "";
    for (index = 0; index < data.ordem.length; index++) {
```

```
        a = a + "<tr style='border-style: dotted'><td style='border-style:
dotted'>" + data.ordem[index].senha + "</td><td style='border-style:
dotted'>" + data.ordem[index].especialidade + "</td><td style='border-style:
dotted'>" + data.ordem[index].nome + "</td></tr>"
    }
    return a;
}

function index() {

    var ger_total = document.querySelector('.ger_total'),
        ger_atual = document.querySelector('.ger_atual'),
        oft_total = document.querySelector('.oft_total'),
        oft_atual = document.querySelector('.oft_atual'),
        car_total = document.querySelector('.car_total'),
        car_atual = document.querySelector('.car_atual'),
        psi_total = document.querySelector('.psi_total'),
        psi_atual = document.querySelector('.psi_atual'),
        ordem = document.querySelector('.ordem'),
        users = document.querySelector('.users'),
        websocket = new WebSocket("ws://" + ip + ":" + porta + "/");
    websocket.onmessage = function (event) {
        data = JSON.parse(event.data);
        som = new sound(data.som);
        switch (data.type) {
            case 'state':

                ger_total.textContent = data.total_geral;
                ger_atual.textContent = data.atual_geral;
                oft_total.textContent = data.total_oftalmologista;
                oft_atual.textContent = data.atual_oftalmologista;
                car_total.textContent = data.total_cardiologia;
                car_atual.textContent = data.atual_cardiologia;
                psi_total.textContent = data.total_psicologia;
                psi_atual.textContent = data.atual_psicologia;
                psi_total.textContent = data.total_psicologia;
                ordem.innerHTML = "<tr style='border-style: dashed'>\n" +
                    "                    <th style='padding-right:
25px;border-style: dashed'>Senha</th>\n" +
                    "                    <th style='padding-right:
25px;border-style: dashed'>Especialidade</th>\n" +
                    "                    <th style='padding-right:
25px;border-style: dashed'>Médico</th>\n" +
                    "                </tr>" + print(data);
                var cor = "#0af";
                for (i = 0; i <= 4; i++) {
                    if (cor == "#0af") {
                        cor = "#a34"
                    } else {
                        cor = "#0af"
                    }
                    ;
                    if ((data.atual - parseInt(i)) > 0) {
                        document.getElementById("fila").innerHTML += "<tr
style='text-align: center; color: " + cor + "'><td style='border-style:
dashed'>" + (data.atual - parseInt(i)) + "</td><td style='border-style:
dashed; text-align: center'>" + data.especialidade + "</td><td style='border-
style: dashed'>" + data.medico + "</td></tr>";
                    }
```

```
                }
                som.play();
                break;
            case 'users':
                users.textContent = (
                    data.count.toString() + " Terminal" +
                    (data.count == 1 ? "" : "s"));
                break;
            default:
                console.error(
                    "unsupported event", data);
        }
    };
}
```

Avançar senhas

A função para avançar a senha é bastante simples constituído em 4 botões que dependendo do botão clicado as informações enviadas ao servidor variam:

```
function senha() {
    var ger_plus = document.querySelector('.ger_plus'),
        oft_plus = document.querySelector('.oft_plus'),
        car_plus = document.querySelector('.car_plus'),
        psi_plus = document.querySelector('.psi_plus'),
        users = document.querySelector('.users'),
        websocket = new WebSocket("ws://" + ip + ":" + porta + "/");
    ger_plus.onclick = function (event) {
        websocket.send(JSON.stringify({action: 'senha', esp: 'ger'}));
    }
    oft_plus.onclick = function (event) {
        websocket.send(JSON.stringify({action: 'senha', esp: 'oft'}));
    }
    car_plus.onclick = function (event) {
        websocket.send(JSON.stringify({action: 'senha', esp: 'car'}));
    }
    psi_plus.onclick = function (event) {
        websocket.send(JSON.stringify({action: 'senha', esp: 'psi'}));
    }
    websocket.onmessage = function (event) {
        data = JSON.parse(event.data);
        if (data.last == "ger") {
            alert("A sua senha é: GER_" + data.total_geral)
        } else if (data.last == "oft") {
            alert("A sua senha é: OFT_" + data.total_oftalmologista)
        } else if (data.last == "car") {
            alert("A sua senha é: CAR_" + data.total_cardiologia)
        } else if (data.last == "psi") {
            alert("A sua senha é: PSI_" + data.total_psicologia)
        }
        switch (data.type) {
            case 'state':
                //value.textContent = data.total;
                break;
            case 'users':
                users.textContent = (
                    data.count.toString() + " Terminal" +
                    (data.count == 1 ? "" : "s"));
                break;
            default:
                console.error(
```

```
                    "unsupported event", data);
        }
    };
}
```

Gestão administrativa

As funções que permitem a um administrador gerir o fluxo de senha são duas sendo as primeira para mostrar as senhas atuais ao utilizador e outra para as atualizar

```
function index_adm() {
    var total = document.querySelector('.total'),
        atual = document.querySelector('.atual'),
        tamanho = document.querySelector('.tamanho'),
        users = document.querySelector('.users'),
        espez = document.cookie.match('especializacao' + '=([^;]*)'),
        websocket = new WebSocket("ws://" + ip + ":" + porta + "/");
    websocket.onmessage = function (event) {
        data = JSON.parse(event.data);
        switch (data.type) {
            case 'state':
                if (espez[1] == 'Geral') {
                    total.textContent = data.total_geral;
                    atual.textContent = data.atual_geral;
                    tamanho.textContent = (data.total_geral -
data.atual_geral);
                } else if (espez[1] == 'Oftalmologista') {
                    total.textContent = data.total_oftalmologista;
                    atual.textContent = data.atual_oftalmologista;
                    tamanho.textContent = (data.total_geral -
data.atual_geral);
                } else if (espez[1] == 'Cardiologia') {
                    total.textContent = data.total_cardiologia;
                    atual.textContent = data.atual_cardiologia;
                    tamanho.textContent = (data.total_cardiologia -
data.atual_cardiologia);
                } else if (espez[1] == 'Psicologia') {
                    total.textContent = data.total_psicologia;
                    atual.textContent = data.atual_psicologia;
                    tamanho.textContent = (data.total_psicologia -
data.atual_psicologia);
                }

                status = data;
                var status = data['status'];
                console.log(status);
                if (status == 'error') {
                    alert('Não há mais senhas na fila');
                }
                break;
            case 'users':
                users.textContent = (
                    data.count.toString() + " Terminal" +
                    (data.count == 1 ? "" : "s"));
                break;
            default:
                console.error(
                    "unsupported event", data);
        }
    };
}
```

```
function adm() {
    var plus = document.querySelector('.plus'),
        value = document.querySelector('.value'),
        users = document.querySelector('.users'),
        espez = document.cookie.match('especializacao' + '=([^;]*)'),
        medico = document.cookie.match('medico' + '=([^;]*)'),
        websocket = new WebSocket("ws://" + ip + ":" + porta + "/");
    console.log(espez[1]);
    plus.onclick = function (event) {
        websocket.send(JSON.stringify({action: 'proximo', esp: espez[1],
"medico": medico[1]}));
    }
    websocket.onmessage = function (event) {
        data = JSON.parse(event.data);
        switch (data.type) {
            case 'state':
                //value.textContent = data.atual;
                break;
            case 'users':
                users.textContent = (
                    data.count.toString() + " Terminal" +
                    (data.count == 1 ? "" : "s"));
                break;
            default:
                console.error(
                    "unsupported event", data);
        }
    };
}
```

Parte Gráfica

A parte gráfica foi construída usando a framework Bootstrap, sendo assim a pouco código com interesse académico, sendo o mais interessante para cada parte o seguinte:

Main Page

```
    <div class="d-flex flex-column"
         style="align-items:center; padding-left: 55px; padding-top: 25px;
border-style: dashed">
        <div style="border-style: solid; font-size: xx-large">
            Quadro Principal
        </div>

        <div class="buttons"
             style="padding-inline: 5px; padding-top: 15px;font-size: 30px;
align-content: center; display: inline-block">
            <div style="border-style: solid; text-align:center;font-size: xx-
large">
                Geral
            </div>
            <a class="value" style="border-style: dashed"> Senhas totais:
                <e class="ger_total">?</e>
            </a>
            <a class="value" style="border-style: solid"> Senha atual:
                <e class="ger_atual">?</e>
            </a>
        </div>

        <div class="buttons"
```

```
            style="padding-inline: 5px; padding-top: 15px;font-size: 30px;
align-content: center; display: inline-block">
            <div style="border-style: solid; text-align:center;font-size: xx-
large">
                Oftalmologista
            </div>
            <a class="value" style="border-style: dashed"> Senhas totais:
                <e class="oft_total">?</e>
            </a>
            <a class="value" style="border-style: solid"> Senha atual:
                <e class="oft_atual">?</e>
            </a>
        </div>

        <div class="buttons"
            style="padding-inline: 5px; padding-top: 15px;font-size: 30px;
align-content: center; display: inline-block">
            <div style="border-style: solid; text-align:center;font-size: xx-
large">
                Cardiologia
            </div>
            <a class="value" style="border-style: dashed"> Senhas totais:
                <e class="car_total">?</e>
            </a>
            <a class="value" style="border-style: solid"> Senha atual:
                <e class="car_atual">?</e>
            </a>
        </div>

        <div class="buttons"
            style="padding-inline: 5px; padding-top: 15px;font-size: 30px;
align-content: center; display: inline-block">
            <div style="border-style: solid; text-align:center;font-size: xx-
large">
                Psicologia
            </div>
            <a class="value" style="border-style: dashed"> Senhas totais:
                <e class="psi_total">?</e>
            </a>
            <a class="value" style="border-style: solid"> Senha atual:
                <e class="psi_atual">?</e>
            </a>
        </div>

        <div class="state">
            <span class="users">?</span> online
        </div>
        <div>
            <table class="ordem" style="font-size: xx-large; border-style:
solid; text-align: center; vertical-align: center" id="fila">


            </table>
            <script> index()</script>
        </div>

    </div>

</div>
```

Avançar senhas

```
<div class="d-flex flex-column" style="text-align: ;: center">
    <div style="font-size: xx-large">
        Clinica Geral
    </div>
    <div class="buttons" style="padding-inline: 5px; padding-top: 15px;
padding-bottom:15px;font-size: 30px;position: relative; align-content:
center; display: inline-block">
        <a class="ger_plus button">Geral</a>
    </div>
    <div class="buttons" style="padding-inline: 5px; padding-top: 15px;
padding-bottom:15px;font-size: 30px;position: relative; align-content:
center; display: inline-block">
        <a class="oft_plus button">Oftalmologista </a>
    </div>
    <div class="buttons" style="padding-inline: 5px; padding-top: 15px;
padding-bottom:15px;font-size: 30px;position: relative; align-content:
center; display: inline-block">
        <a class="car_plus button">Cardiologia</a>
    </div>
    <div class="buttons" style="padding-inline: 5px; padding-top: 15px;
padding-bottom:15px;font-size: 30px;position: relative; align-content:
center; display: inline-block">
        <a class="psi_plus button">Psicologia</a>
    </div>
    <div class="state">
        <span class="users">?</span> online
    </div>
    <script>senha();</script>
</div>
```

Dashboard Medico

O seguinte código vai definir qual a especialidade do médico

```
<?php
session_start();
if (isset($_SESSION["user"])) {
    echo "<li class='nav-item medico' role='presentation'
onclick='logout()'><a class='nav-link' ><i class='fas fa-sign-out-
alt'></i><span >Logout</span></a></li>";
}
?>
<script type="text/javascript">
    document.cookie = "especializacao=<?php echo $user['especializacao'] ?>";
    document.cookie = "medico=<?php echo $user['nome']." ".$user['apelido']
?>";
</script>
```

O seguinte código é o que vai criar dashboard do médico

```
<div style="align-items:center; padding-left: 55px; padding-top: 25px;
padding-bottom: 25px; text-align: left">
    <div style="font-size: 50px; font-weight: bold; text-align: left"
class="medico">
       <?= 'Médico: ' . $user['nome'] . ' ' . $user['apelido'] ?>
    </div>
    <br>
    <div style="font-size: 50px; font-weight: bold; text-align: left">
        <?= 'Especialidade: <a id="especializacao"> ' .
$user['especializacao'].'</a>' ?>
    </div>
```

```
    <hr>
    <div>
        <table style="font-size: 45px; align-content: center">
            <tr>
                <th style="border-style:dashed; padding-right: 20px; padding-
left: 20px; text-align: center ">Total de senhas</th>
                <th style="border-style:dashed; padding-left: 20px; padding-
right: 20px; text-align: center">Senha atual</th>
            </tr>
            <tr>
                <td style="border-style:dashed; padding-right: 20px; text-
align: center " class="total">?</td>
                <td style="border-style:dashed; padding-left: 20px; text-
align: center" class="atual">?</td>
            </tr>
            <tr>
                <th colspan="2" style="border-style: dashed; text-align:
center;">Tamanho da fila</th>
            </tr>
            <tr>
                <td colspan="2" style="border-style: dashed; text-align:
center" class="tamanho">?</td>
            </tr>
            <tr>
                <td colspan="2" style="border-style: dashed; text-align:
center;background-color: #4e73df;" class="plus"><a style="color:
black">+</a></td>
            </tr>
        </table>
    </div>
    <script>adm()</script>
    <footer style="text-align: center" class="state">
        <span style="text-align: center" class="users">?</span> online
        <script>
            status = index_adm();
        </script>
    </footer>
</div>
```

Login

```
<?php
require_once '../DB_config.php';
session_start();
session_destroy();
session_start();
if($_POST)
{

    $user_email    = $_POST['email'];
    $user_password     = $_POST['pass'];


    //password_hash see : http://www.php.net/manual/en/function.password-
hash.php
    $password  = hash('sha512', $user_password);
    try
    {
        $stmt = $db_con->prepare("SELECT * FROM medicos WHERE
email='".$user_email."' and pass='".$password."'");
        $stmt->execute();
```

```
        $count = $stmt->rowCount();

        if($count==1){
            foreach ($stmt as $row) {
                $_SESSION["user"] = $row;
            }
                echo "logado";
                header("Location: /python_web/adm");
        }

            else
            {
                echo "<br>Email ou Pass incorreta<br>";
            }


    }
    catch(PDOException $e){
        echo $e->getMessage();
    }
}
else{
    header("Location: /python_web/adm/");
}
?>
```

Registo

```
<?php
require_once '../../DB_config.php';

if($_POST)
{
    $user_nome        =  $_POST['nome'];
    $user_apelido     = $_POST['apelido'];
    $user_email    = $_POST['email'];
    $user_especialidade    = $_POST['especialidade'];
    $user_password     = $_POST['pass'];
    $joining_date  = date('Y-m-d H:i:s');

    //password_hash see : http://www.php.net/manual/en/function.password-
hash.php
    $password  = hash('sha512', $user_password );
    try
    {
        $stmt = $db_con->prepare("SELECT * FROM medicos WHERE email=:email");
        $stmt->execute(array(":email"=>$user_email));
        $count = $stmt->rowCount();

        if($count==0){
            $stmt = $db_con->prepare("INSERT INTO
medicos(nome,apelido,pass,email,especializacao,registo) VALUES(:nome,
:apelido, :pass, :email, :especialidade, :jdate)");
            $stmt->bindParam(":nome",$user_nome);
            $stmt->bindParam(":apelido",$user_apelido);
            $stmt->bindParam(":email",$user_email);
            $stmt->bindParam(":especialidade",$user_especialidade);
            $stmt->bindParam(":pass",$password);
            $stmt->bindParam(":jdate",$joining_date);
            if($stmt->execute())
            {
```

```
                echo "registered";
                header("Location: /python_web/adm/");
            }
            else
            {
                echo "Query could not execute !";
            }

        }
        else{

            echo "1"; //  not available
        }

    }
    catch(PDOException $e){
        echo $e->getMessage();
    }
}
else{
    header("Location: /python_web/adm/reg/");
}
?>
```

Produto Final

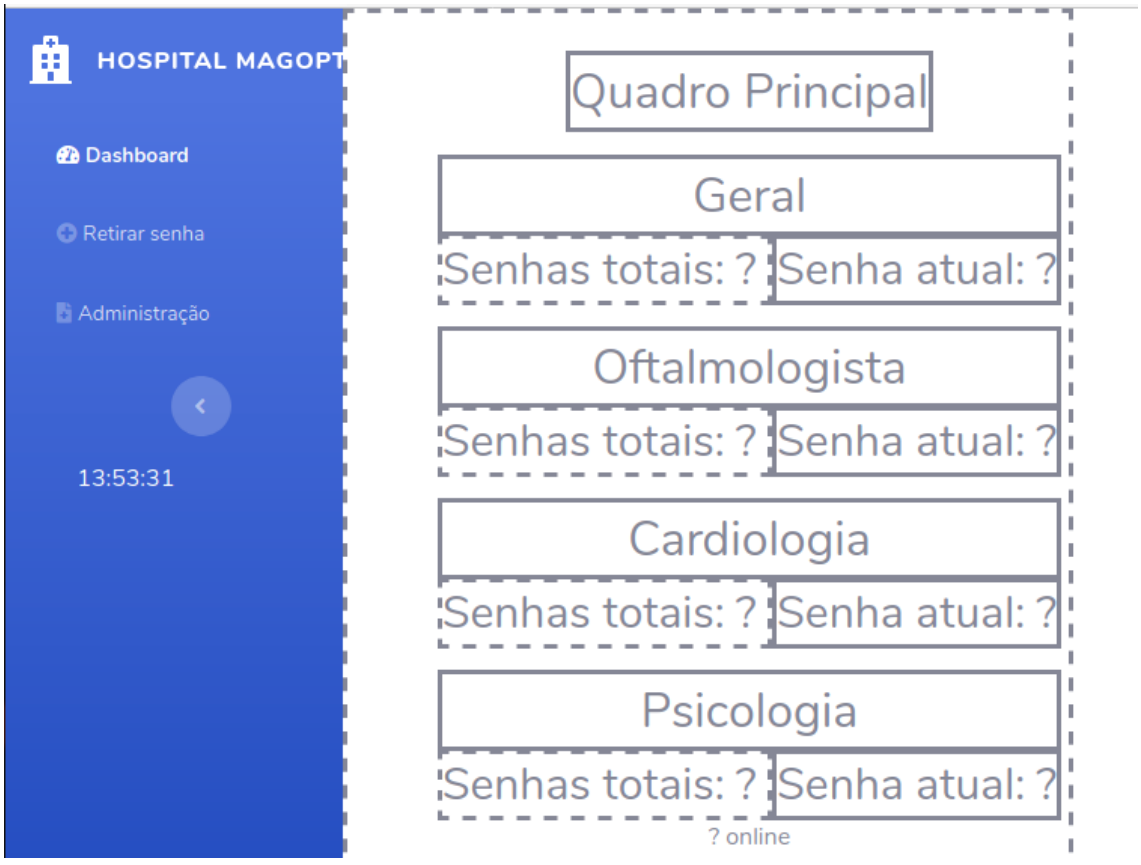


Figura 1- Main Page

Figura 2- Senhas

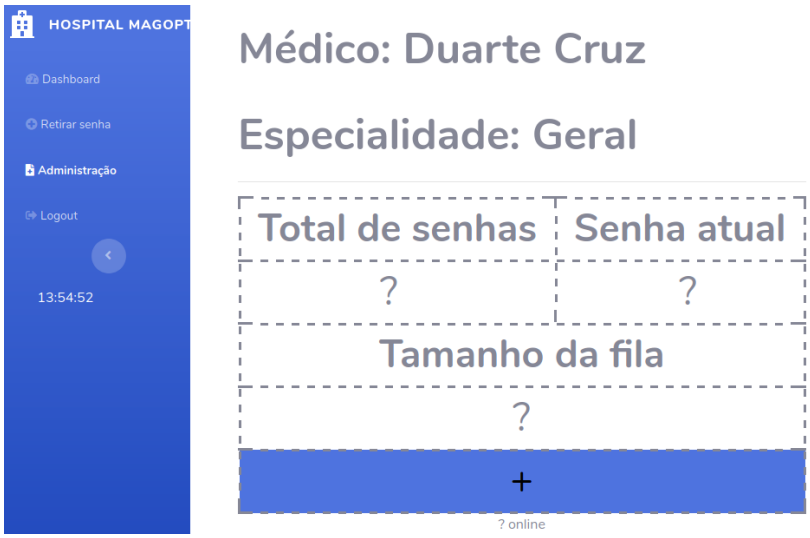


Figura 3- Administração

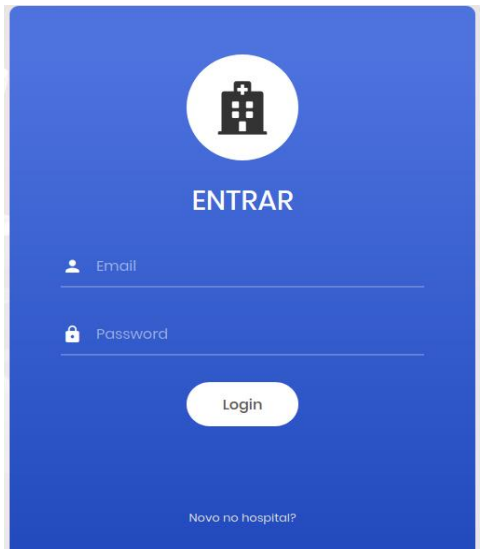


Figura 4- Login

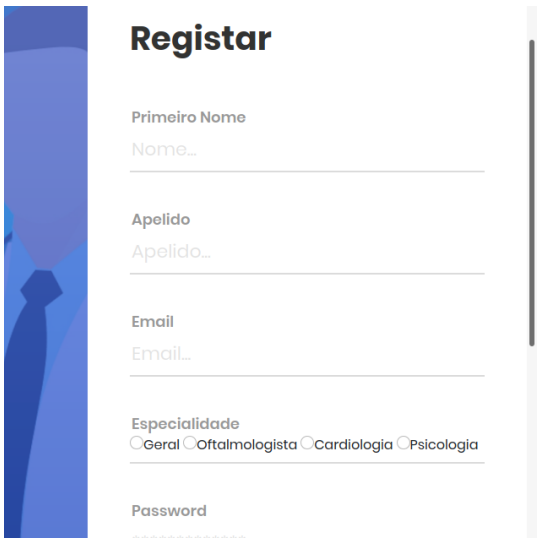


Figura 5 – Registar

Web Grafia

Bootstrap Studio: https://bootstrapstudio.io/

Ubuntu: https://ubuntu.com/

Python: https://www.python.org/

Sockets Python: https://websockets.readthedocs.io/en/stable/intro.html