**6) Relatório das Linhas de Códigos**

Como a aplicação consta com inúmeras funcionalidades, neste tópico será englobado apenas os procedimentos de captura e armazenamento da foto do usuário e do reconhecimento facial, sendo os subtópicos:

**- alteracao-biometria.jsp;**

**- iniciarFotoCapturaReconhecimento.js;**

**-** **ProcessamentoAlterarImagemReconhecimento.java;**

**- UsuarioDao.AlterarFotoReconhecimento.java**

**- reconhecimento.html;**

**- reconhecimento.js**.

**6.1) alteracao-biometria.jsp**

Página web onde carrega os módulos de captura da imagem do usuário via webcam.

<%@ **page** language=*"java"* contentType=*"text/html; charset=utf8"*

pageEncoding=*"utf8"*%>

<%@**page** import=*"Seguranca.StripHTML"* %>

<!**DOCTYPE** html>

<**html** lang=*"pt-br"*>

<**jsp:include** page=*"Include\head-login.jsp"* />

<**body**>

<**jsp:include** page=*"Include\header.jsp"* />

<**section** class=*"container flexCenter"*>

<**div** class=*"formBiometria neomorphismo uiFade flexCenter"*>

<**br**><**br**>

<**jsp:include** page=*"Include\titulo.jsp"* />

<**br**><**br**>

<!-- mensagem de informação -->

<**p** class=*"mensagemInformacao"*>🎉 Vamos começar com a alteração, clique no primeiro botão abaixo!</**p**>

<**br**><**br**>

<**button** id=*"iniciarCaptura"* class=*"fas fa-camera campoRedondo neomorphismo neonHover opacito uiFade"*></**button**>

<**br**><**br**>

<!-- elementos da captura da face -->

<**div** id=*"camera"*></**div**>

<**div** class=*"separador"*></**div**>

<**button** id=*"capturarFoto"* class=*"fas fa-camera campoRedondo neomorphismo neonHover opacito uiFade"*></**button**>

<**div** class=*"separador"*></**div**>

<**img** id=*"previaImagemCapturada"* width=*"240px"* height=*"240px"*>

<**div** class=*"separador"*></**div**>

<**div** id=*"formulario"*></**div**>

</**div**>

</**section**>

<**jsp:include** page=*"Include\footer.jsp"* />

</**body**>

<**jsp:include** page=*"Include\script-login.jsp"* />

<**jsp:include** page=*"Include\Resetar Sessoes\Resetar-Sessoes-Geral.jsp"* />

</**html**>

**6.2) iniciarFotoCapturaReconhecimento.js**

Script que gerencia a funcionalidade de captura da imagem do usuário via webcam.

/\*

\* capturação da foto a ser utilizada no reconhecimento facial

\* \*/

// ocultação dos elementos utilizados somente na captura e captura de variáveis globais

$(document).ready(function() {

// ocultação

$("#camera").css("display", "none");

$(".separador").css("display", "none");

$("#capturarFoto").css("display", "none");

$("#previaImagemCapturada").css("display", "none");

$("#formulario").css("display", "none");

});

// inicialização da câmera e do processo de captura

$("#iniciarCaptura").click(function() {

// retira dados da página

$(".tituloSection").css("display", "none");

$(".mensagemInformacao").css("display", "none");

$("#iniciarCaptura").css("display", "none");

// reposicionamento do painel para flex-direction row

$(".formBiometria").css("flex-direction", "row");

// adiciona conteúdos para captura da face

$("#camera").css("display", "block");

$(".separador").css("display", "block");

$("#capturarFoto").css("display", "block");

// preparação da webcam

Webcam.set({

// tamanho da janela de visualização

width: 320,

height: 240,

// tamanho da área capturada

dest\_width: 640,

dest\_height: 480,

// tamanho da área capturada considerado

crop\_width: 480,

crop\_height: 480,

// formato da imagem

image\_format: 'jpeg',

jpeg\_quality: 100

// girar imagem para ficar posicionada do jeito real

//flip\_horiz: true

});

Webcam.attach('#camera');

});

// captura e visualização da preview

$("#capturarFoto").click(function() {

Webcam.snap(function(data\_uri) {

// exibição do resultado da captura

$("#previaImagemCapturada").attr("src", data\_uri);

$("#previaImagemCapturada").css("display", "block");

// preparação do formulário

$("#formulario").css("display", "block");

$("#formulario").html('<form method="POST" action="ProcessamentoAlterarImagemReconhecimento" class="modal-content neomorphismo" enctype="multipart/form-data">' +

'<input name="uuid\_usuario" type="hidden" value="' + window.localStorage.getItem("uuid\_usuario") + '">' +

'<input name="uuid\_alterar\_biometria" type="hidden" value="' + window.localStorage.getItem("uuid\_alterar\_biometria") + '">' +

'<input name="fotoReconhecimento" type="hidden" value="' + data\_uri + '"/>' +

'<input type="submit" class="botao neomorphismo opacito neonHover" value="Enviar">' +

'</form>');

});

});

**6.3) ProcessamentoAlterarImagemReconhecimento.java**

Servlet do lado do servidor responsável por receber a imagem captura e armazenar no banco de dados.

package Servlet;

import java.awt.image.BufferedImage;

import java.io.ByteArrayInputStream;

import java.io.File;

import java.io.IOException;

import java.io.InputStream;

import javax.imageio.ImageIO;

import javax.servlet.ServletException;

import javax.servlet.annotation.MultipartConfig;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.xml.bind.DatatypeConverter;

import Bean.AlterarBiometriaBean;

import Dao.AlterarBiometriaDao;

import Dao.UsuarioDao;

*@WebServlet*(urlPatterns = "/ProcessamentoAlterarImagemReconhecimento")

*@MultipartConfig*(maxFileSize = 16177215) // processa imagens até 16 MB (valor informado em B)

public class ProcessamentoAlterarImagemReconhecimento extends HttpServlet {

/\*

\* Classe responsável por receber e tratar o envio de imagens de reconhecimento pelos usuários

\* Após receber, ela irá descriptar as partes do arquivo, uni-las em um único pacote

\* e encaminhar ao banco de dados

\* \*/

private static final long ***serialVersionUID*** = 4L;

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

/\*

\* Processamento de envio da imagem em método POST

\* Assintótica: O(1)

\* \*/

InputStream foto\_reconhecimento = null;

// captura da foto do usuário e conversão em bytes

String foto\_usuario = request.getParameter("fotoReconhecimento");

foto\_usuario = foto\_usuario.split(",")[1];

byte[] foto\_usuario\_bytes = DatatypeConverter.*parseBase64Binary*(foto\_usuario);

String uuid\_usuario = request.getParameter("uuid\_usuario");

// Caso a imagem foi encaminhada com sucesso à Servlet

// Tal será processada e inserida no banco de dados

if (foto\_usuario != null) {

// alterando foto no registro do usuário

foto\_reconhecimento = new ByteArrayInputStream(foto\_usuario\_bytes);

UsuarioDao usuario\_dao = new UsuarioDao();

usuario\_dao.AlterarFotoReconhecimento(uuid\_usuario, foto\_reconhecimento);

// marcando link da biometria como usado e expirado

AlterarBiometriaBean alterar\_biometria = new AlterarBiometriaBean();

AlterarBiometriaDao alterar\_biometria\_dao = new AlterarBiometriaDao();

alterar\_biometria.setUuidAlterarBiometria(request.getParameter("uuid\_alterar\_biometria"));

alterar\_biometria\_dao.UsarAlteracao(alterar\_biometria);

// salvando foto do usuário no servidor

try {

String nickname = usuario\_dao.SelecionarNickname(uuid\_usuario);

ByteArrayInputStream bais = new ByteArrayInputStream(foto\_usuario\_bytes);

BufferedImage bImageFromConvert = ImageIO.*read*(bais);

System.***out***.println(nickname);

System.***out***.println("\n\n---------\n\n");

System.***out***.println(bais);

System.***out***.println("\n\n---------\n\n");

System.***out***.println(bImageFromConvert);

System.***out***.println("\n\n---------\n\n");

ImageIO.*write*(bImageFromConvert, "png", new File("C:\\Users\\gabri\\Desktop\\Faculdade\\APS\\Reconhecimento Facial\\assets\\imagensReconhecimento\\" + nickname + ".png"));

System.***out***.println("Images were written succesfully.");

}

catch (IOException e) { System.***out***.println("Exception occured :" + e.getMessage()); }

}

response.sendRedirect("biometria-alterada.jsp");

}

}

**6.4) UsuarioDao.AlterarFotoReconhecimento.java**

Método da classe UsuarioDao responsável por armazenar a foto de reconhecimento do usuário no banco de dados MySQL.

[…]

public boolean AlterarFotoReconhecimento(String uuid\_usuario, InputStream imagem) {

try {

String sql = "UPDATE usuario SET"

+ " foto\_reconhecimento = ?"

+ " WHERE uuid\_usuario = UUID\_TO\_BIN(?);";

Connection conexao = ConexaoMySQL.*AbrirConexao*();

PreparedStatement smtp = conexao.prepareStatement(sql);

smtp.setBlob(1, imagem);

smtp.setString(2, uuid\_usuario);

smtp.executeUpdate();

smtp.close();

ConexaoMySQL.*FecharConexao*(conexao);

return true;

}

catch (SQLException e) {

e.printStackTrace();

return false;

}

}

[…]

**6.5) reconhecimento.html**

Página web de aplicação secundária responsável por carregar os algoritmos de reconhecimento facial bem como ativar a webcam do usuário para dar início ao processo de reconhecimento.

<html>

<head>

<title>..:: RecognizeMe! ..::</title>

<!-- Metas -->

<meta charset="utf-8">

<meta name="author" content="@CSFelix, @CSVini & @CSAugustera">

<meta name="keywords" content="face, face-detection, face- recognition, artificial-inteligence">

<meta name="description" content="Detect and Recognize Users' Faces">

<meta name="viewport" content="width=device-width,initial- scale=1,maximum-scale=1,user-scalable=no">

<link rel="stylesheet" href="./assets/css/variaveis.css">

<link rel="stylesheet" href="./assets/css/estilo- reconhecimento.css">

</head>

<body>

<video id="cam"

width="320"

height="240"

autoplay

muted>

</video>

</body>

<script src="./assets/lib/face-api/face-api.min.js"></script>

<script src="./assets/js/reconhecimento.js"></script>

</html>

**6.6) reconhecimento.js**

Script responsável por detectar e reconhecer faces dos usuários, bem como o redirecionamento para liberação total ou não do acesso à aplicação.

// objeto que gerencia a câmera do usuário e exibe a imagem do mesmo

const cam = document.getElementById("cam");

// dados para comparação final

const queryString = window.location.search;

const urlParams = new URLSearchParams(queryString);

const nickname\_usuario = urlParams.get("label");

const acuracia\_meta = 0.01;

// resultados obtidos pela análise

var acuracia\_final = 0;

var label\_final;

var dados = [,];

alert("A página pode demorar um pouco para carregar, espere até ela desenhar pontos em seu rosto.");

alert("Ao desenhar os pontos, nossa Assistente Virtual demorará certa de 60 segundos para analisar seu rosto."

+ " Passado o tempo, você será redirecionado para o painel de notícias automaticamente.");

/\* reconhecimento e carregamento da câmera do usuário \*/

const startVideo = () => {

// lista dos dispositivos de entrada\saída

navigator.mediaDevices.enumerateDevices()

.then(devices => {

// checar se encontrou dispositivos

if (Array.isArray(devices)) {

// filtragem pelo dispositivo de vídeo

devices.forEach(device => {

if (device.kind === 'videoinput') {

if (device.label.includes('')) {

// acesso à câmera

navigator.getUserMedia(

{ video: { deviceId: device.deviceId }},

stream => cam.srcObject = stream,

error => console.error(error)

);

}

}

});

}

});

}

/\* carregamento da label para reconhecimento \*/

const loadLabels = () => {

const labels = [nickname\_usuario];

return Promise.all(labels.map(async label => {

const descriptions = [];

for (let i = 1; i <= 1; i++) {

const img = await faceapi.fetchImage(`/assets/imagensReconhecimento/${nickname\_usuario}.png`);

const detections = await faceapi

.detectSingleFace(img)

.withFaceLandmarks()

.withFaceDescriptor()

descriptions.push(detections.descriptor);

}

return new faceapi.LabeledFaceDescriptors(label, descriptions);

}));

}

/\* modelos da face api carregadas para reconheceram e detectarem faces \*/

Promise.all([

// detecta faces\rostos

faceapi.nets.tinyFaceDetector.loadFromUri("/assets/lib/face-api/models"),

// detecta os traços dos rostos (os 128 pontos)

faceapi.nets.faceLandmark68Net.loadFromUri("/assets/lib/face-api/models"),

// reconhece faces\rostos

faceapi.nets.faceRecognitionNet.loadFromUri("/assets/lib/face-api/models"),

// utilizada em segundo plano para detecção do rosto

faceapi.nets.ssdMobilenetv1.loadFromUri("/assets/lib/face-api/models")

]).then(startVideo);

/\* exibição do vídeo na página e toda a lógica da detecção\reconhecimento da face \*/

cam.addEventListener('play', async () => {

// desenho do canvas com atualização dinâmica das imagens

const canvas = faceapi.createCanvasFromMedia(cam);

const canvasSize = { width: cam.width, height: cam.height }

const labels = await loadLabels();

faceapi.matchDimensions(canvas, canvasSize);

document.body.appendChild(canvas);

setInterval(async () => {

const detections = await faceapi

.detectAllFaces(cam, new faceapi.TinyFaceDetectorOptions())

.withFaceLandmarks()

.withFaceDescriptors();

const resizedDetections = faceapi.resizeResults(detections, canvasSize)

const faceMatcher = new faceapi.FaceMatcher(labels, acuracia\_meta)

const results = resizedDetections.map(d =>

faceMatcher.findBestMatch(d.descriptor))

// reset do canvas

canvas.getContext('2d').clearRect(0, 0, canvas.width, canvas.height);

// detecção da face

faceapi.draw.drawDetections(canvas, resizedDetections);

// detecção dos pontos do rosto

faceapi.draw.drawFaceLandmarks(canvas, resizedDetections);

// reconhecimento de faces

results.forEach((result, index) => {

const box = resizedDetections[index].detection.box;

const { label, distance } = result;

new faceapi.draw.DrawTextField([

//`${label} (${parseInt(distance \* 100, 10)})`

`${nickname\_usuario} (${parseInt(distance \* 100, 10)})`

], box.bottomRight).draw(canvas);

//dados[0] = label;

dados[0] = nickname\_usuario;

dados[1] = distance;

});

distancia\_obtida = parseInt(dados[1] \* 100, 10);

label\_final = dados[0];

acuracia\_final = distancia\_obtida > acuracia\_final ?

distancia\_obtida

: acuracia\_final;

}, 100);

setTimeout(function(){

// reconhecido

if (label\_final == nickname\_usuario && acuracia\_final >= parseInt(acuracia\_meta \* 100, 10)) {

window.location.replace("http://localhost:8080/DataMe/painel.jsp?tipo\_acesso=1");

}

// não-reconhecido

else {

window.location.replace("http://localhost:8080/DataMe/painel.jsp?tipo\_acesso=0");

}

}, 30000);

});