Projeto de Sistemas Embarcados
Sistema de controle de acesso de usuários integrado a nuvem

Angélica Kathariny de Oliveira Alves, Josué Bezerra Bonfim Filho

***Resumo*—De acordo com os requisitos necessários para o controle de acesso ao Laboratório de Engenharia e Inovação (LEI), este artigo tem como objetivo desenhar um produto cuja função principal é realizar o gerenciamento de acesso dos integrantes do LEI, com integração com e-mail e nuvem. O documento explicita o projeto de hardware e software do sistema, como é feita a coleta e o processamento de dados e o uso de API's de integração com sistemas externos. O sistema faz uso de uma Raspberry Pi Modelo B, um leitor de biometria e um display LCD 16x2.**

***Keywords*—Raspberry-pi, controle-de-acesso, biometria, API, e-mail**

I. INTRODUÇÃO

Um sistema de controle de acesso é capaz de gerenciar e controlar o fluxo de pessoas em áreas restritas. São amplamente utilizados em prédios residenciais e empresariais, academias, universidades e escolas.

Em muitos desses locais seu uso vai além de permitir ou não a entrada de um indivíduo em um determinado recinto. Em geral são utilizados para controle de frequência, cálculo de horas de trabalho, contagem precisa do número de usuários de um ambiente. Esses dados são importantes para uma boa administração do local onde o sistema foi implantado.

O sistema pode ser acionado por meio de cartões de acesso, autenticação por código numérico, leitura de código de barras ou reconhecimento biométrico. A maioria desses sistemas possui fácil instalação e operação do usuário porém possuem baixo nível de segurança, uma vez que as informações podem ser repassadas a terceiros sem autorização da equipe gestora do controle.

Com o intuito de elevar o nível de segurança dos sistemas o uso de reconhecimento biométrico vem sendo a melhor opção a ser implantada. Para isso a liberação de acesso é feita após a leitura e validação de características físicas particulares de cada indivíduo. Os métodos mais utilizados são leitura biométrica da mão, leitura da íris, leitura de impressão digital e reconhecimento facial.

O uso dessas tecnologias ainda propiciam integração com outros sistemas por intermédio softwares que facilitam a gestão e customização de acordo com as necessidades do gestor. Ademais as tecnologias de computação em nuvem e internet das coisas fazem com que essa gestão de informações seja feita remotamente e em tempo real, facilitando a tomada de decisões.

II. JUSTIFICATIVA

O Laboratório de Engenharia e Inovação - LEI representa um núcleo de laboratórios de pesquisa com a missão de produzir, desenvolver e difundir conhecimentos de Engenharias com responsabilidade social, transparência, inovação, ética e multidisciplinaridade. O objetivo geral do LEI é prover um espaço para a consolidação de pesquisa aplicada na área de engenharia e inovação tecnológica no ambiente acadêmico.

Este núcleo é composto por seis laboratórios de pesquisa:

- Laboratório de Bioengenharia e Biomateriais-BioEngLab;
- Laboratório de Gerenciamento de Sistemas Dinâmicos;
- Laboratório de Computação Musical e Acústica;
- Laboratório de Estatística Aplicada à Probabilidade - LEAP;
- Laboratório de Instrumentação e Processamento de Imagens e Sinais - LIPIS;
- Laboratório de Informática e Saúde - LIS;

O LEI integra, aproximadamente, 20 (vinte) pesquisadores e 100 (cem) alunos de graduação e pós-graduação que realizam pesquisa nas áreas de engenharia biomédica, biomateriais, mecânica, eletroeletrônica, engenharia de *Software*, informática em saúde, modelagem matemática e sistemas de controle. Possui diversos recursos tecnológicos como equipamentos de medição e análise, interfaces de interação humano-computador e bancadas de trabalho.

Para promover segurança aos usuários e ao patrimônio do LEI é necessário o uso de um sistema de controle de acesso. O mesmo já está implementado no LEI com o Dispositivo de Controle de acesso 5YBX6. O problema é que esta solução não possui muitos recursos de gerenciamento de usuários e não é integrado a uma rede de comunicação. Sua configuração permite apenas um administrador, cadastro de 150 (cento e cinquenta) usuário além de não possuir base de dados com informações dos usuários, apenas associa a impressão digital cadastrada a um número de identificação. Além disso, a solução tem apresentado problema na validação das impressões digitais cadastradas, bloqueando o acesso de usuários.

III. OBJETIVO

O objetivo desse projeto é desenvolver um sistema de controle de acesso para o laboratório LEI que possua conectividade web. O produto trará benefícios para a coordenação do LEI, como armazenamento de informação dos usuários em nuvem, controle de horário de acesso e a inclusão e

exclusão de novos membros no sistema. A integração com API (*Application Programming Interface*) de e-mail também trará facilidade na comunicação com os usuários dos laboratórios.

IV. Requisitos

O projeto possui os seguintes requisitos:

- Comunicação efetiva entre o leitor biométrico de impressão digital e a *Raspberry Pi*;
- Integração entre a *Raspberry Pi* e a nuvem para o armazenamento das informações dos usuários;
- Integração entre a *Raspberry Pi* e a API de email para a comunicação entre usuários do LEI.

V. Descrição de Hardware

Este projeto fará uso dos seguintes componentes para a solução de hardware:

A. Raspberry Pi 3 Model B

A *Raspberry Pi 3 Model B*, como mostra a FIG. 1, possui custo médio de R$ 200,00. Esta placa possui quatro portas USB (*Universal Serial Bus*), uma porta HDMI *High-Definition Multimedia Interface* para conexão com *display* e um *slot* de cartão microSD para armazenamento de dados, uma vez que a mesma não possui memória interna.

Figura 1. Raspberry Pi 3 Model B

Além disso, possui uma porta para conexão *Ethernet* com velocidades de 10/100 Mbps e *Wi-Fi* para conexão com a *internet*. O sistema operacional usado é o Raspbian (Debian wheezy). A Raspberry Pi 3 é alimentada com uma fonte de 5V e 3A (4.0 W) [7], possui 1GB RAM (*Random Access Memory* e CPU (Unidade Central de Processamento) com velocidade de 900 MHz *quad-core* ARM Cortex-A7.

B. Leitor de Impressão Digital - FPM10A

O projeto conta com um módulo de reconhecimento e gravação de impressão digital, FIG. 2. O módulo escolhido foi o FPM10A genérico fabricado na China. É alimentado por uma tensão contínua que pode variar entre 3,3 e 6,0V, possui uma corrente de funcionamento menor que 120mA, e resiste a uma corrente de pico de até 140mA. Possui uma tela para captura de imagem de 14x18mm e leva aproximadamente um segundo para captar a impressão digital. Sua comunicação com outros dispositivos se dá por meio da interface UART (*Universal asynchronous receiver/transmitter*).Este módulo tem capacidade de armazenamento de 1000 impressões digitais em sua memória e seu microprocessador faz a busca das impressões cadastradas em um segundo.

Figura 2. Leitor de Impressão Digital - FPM10A

C. Display LCD 16x2

O display LCD (display de cristal líquido) é utilizado para informar o usuário quanto ao início e finalização dos processos de cadastro e busca de impressões digitais, informando também se o acesso é negado ou permitido. O modelo usado é o ITM-1602BSTL da INTECH LCD GROUP, ilustrado na FIG. 9.

O modelo possui os pinos um e dois para alimentação do *LED BACKLIGHT+* e *LED BACKLIGHT-* respectivamente. Os pinos três e quatro são para alimentação do display. O pino cinco é usado para controlar o contraste do display e é ligado a um potenciômetro para esse ajuste. O pino seis é o RS (*Register Select*). O pino sete é o *Read/Write select* e o mesmo deve ser ligado ao terminal de referência do circuito. O pino oito é o pino de *Enable* do *Read/Write*. Os pinos de nove a 16 são os *DATA BUS*. Para este projeto, a comunicação será feita

Figura 3. LCD ITM-1602BSTL - INTECH LCD GROUP

por *nibble*, então apenas quatro pinos serão utilizados (de 13 a 16).

VI. Descrição de Software

A. Menu principal

O menu principal apresenta opções para uso do administrador sistema, onde é possível adicionar um novo usuário, editar ou excluir um usuário já cadastrado. A FIG. 4 apresenta o fluxograma do código proposto para este menu.

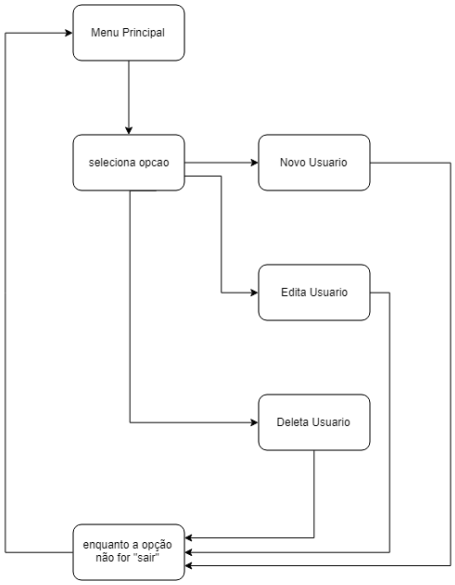

Figura 4. Fluxograma do menu principal do *software*.

Quando o administrador seleciona a opção desejada o *software* faz a validação da mesma e só então acessa a opção desejada, FIG. 5.

B. Cadastro de usuário

Quando o administrador opta por cadastrar um novo usuário devem ser inseridas informações como nome, data de nascimento e e-mail. A FIG. 6 apresenta o fluxograma do cadastros do usuário, onde mostra que as informações inseridas são armazenadas em um arquivo próprio para esse usuário. Em seguida esse arquivo é armazenado junto aos arquivos dos usuários já cadastrados.

C. Cadastro da impressão digital

De acordo com o apresentado na FIG. 7, quando se deseja cadastrar a impressão digital de um usuário o display apresenta as instruções necessárias depois de estabelecida a comunicação com o sensor biométrico. O usuário deve pressionar o sensor duas vezes para o armazenamento da digital e, ao pressionar pela terceira vez, uma imagem da digital é armazenada ao arquivo que contém as informações referentes ao mesmo.

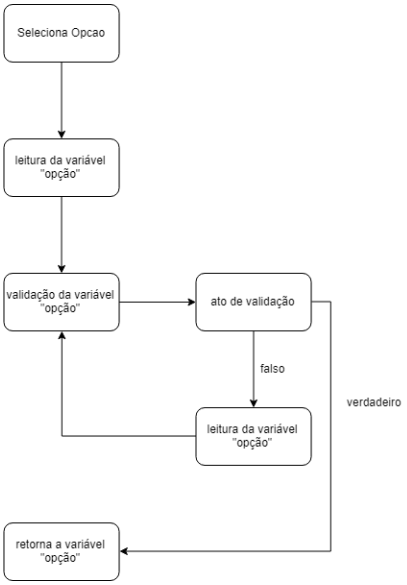

Figura 5. Validação da opção selecionada no meu principal.

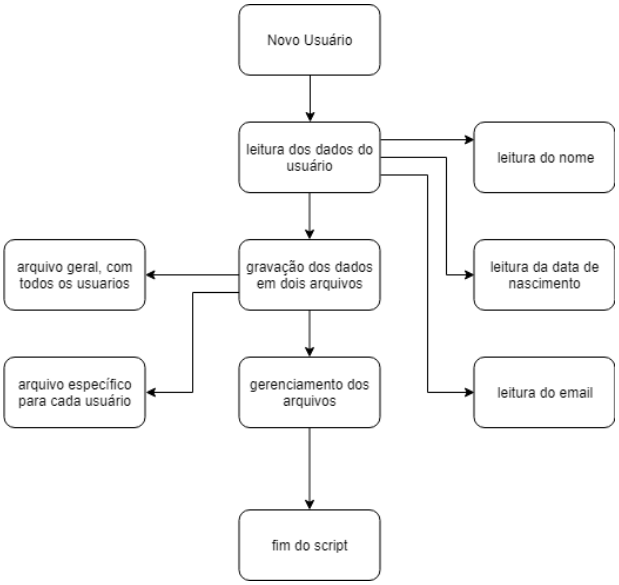

Figura 6. Fluxograma do cadastro de um usuário.

D. Upload no Dropbox

Ao final do processo de cadastramento das informações do usuário o *software* envia os arquivos para armazenamento em nuvem no Dropbox,onde somente o administrador do sistema tem acesso as informações. Caso seja necessário fazer atualização de informações o arquivo é atualizado ao final do processo.

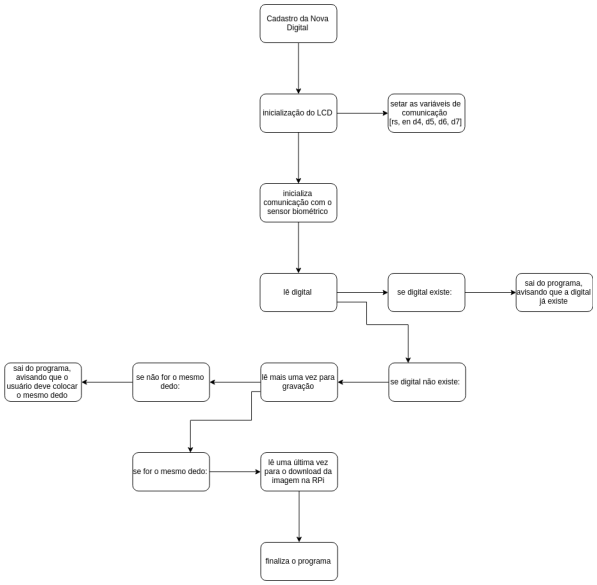

Figura 7. Processo para cadastro da impressão digital de um usuário.

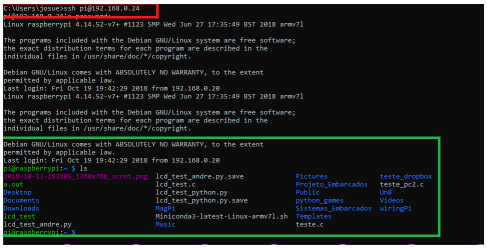

Figura 8. Teste da Conexão RPi - OS (Windows ou Linux)

VII. RESULTADOS

A. Comunicação RaspberryPi - Linux

A figura 8 mostra o prompt de comando do Windows 10 conectado com a RPi. Para que isso aconteça, precisamos encontrar o ip da RPi. Uma vez encontrado, e sabendo que o Windows 10 possui um cliente de ssh nativo, basta executar `> ssh pi@ip_da_RPi` aceitar a conexão e digitar a senha da RPi para que a conexão seja estabelecida.

Pode-se observar que o retângulo vermelho representa a etapa de conexão entre o Windows 10 e a RPi, enquanto o retângulo verde representa a etapa do controle total do terminal da RPi. Logo, pode-se concluir que a comunicação ocorreu da forma esperada.

B. Comunicação RaspberryPi - LCD 16x2

A figura 9 mostra o resultado do teste da comunicação RPi - LCD 16x2. Através de um script em Python, foi possível enviar a data e a string "Teste Embarcados"para o display, que exibiu a mensagem conforme esperado.

C. Comunicação RPi - Dropbox

As figuras 10 e 11 mostram o resultado do teste da comunicação RPi - Dropbox.

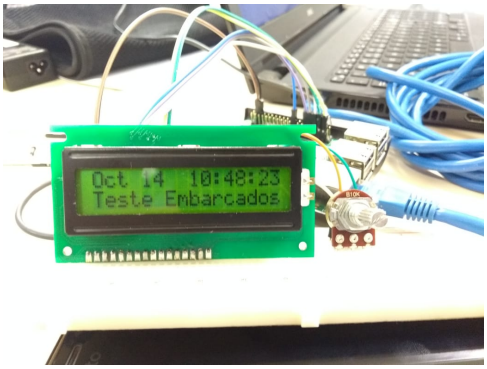

Figura 9. Teste da Conexão RPi - LCD 16x2

Figura 10. Teste da Conexão RPi - Dropbox no terminal

Pode-se observar que a comunicação ocorreu de forma esperada.

D. Comunicação RPi - FPM10A

O teste foi realizado da seguinte forma: Um dedo não cadastrado no sistema foi colocado na tela do sensor. O mesmo leu a digital e informou que não há qualquer cadastro deste usuário. Então, o sistema cadastra a digital na base de dados. O mesmo dedo que foi cadastrado foi recolocado na leitura do sistema, que retornou o número no qual a impressão digital foi cadastrada.

A figura 12 mostra o resultado deste teste, onde o retângulo vermelho indica a leitura do dedo não cadastrado, o retângulo azul indica o cadastro e o retângulo verde mostra que a digital foi cadastrada e foi reconhecida pela busca. A partir disso, afirma-se que o teste ocorreu de forma esperada.

E. Display de interface com o usuário

Foi realizado um teste com o display LCD para exibir as mensagens apresentadas ao usuário. A tela inicial é apresentada ao usuário é mostrada na na fig.13.

Ao tentar o acesso ao laboratório, se o usuário já estiver cadastrado, o display informará a mensagem "Usuário, seja bem-vindo, conforme mostra a fig. 14

Caso o usuário não esteja cadastrado esta mensagem é apresentada no display e seu acesso é negado ao laboratório.

VIII. CONSIDERAÇÕES FINAIS

Ao contemplar o material exposto neste documento, é possível notar que a base teórica para o projeto está consolidada e se mostra como um ponto de partida concreto para o desenvolvimento do sistema. Por meio de artigos, é possível notar que outras soluções com finalidades semelhantes a proposta nesse trabalho porém sem a comunicação necessária para o gerenciamento de usuários requerido pelo laboratório LEI. Com base no tempo e no escopo do projeto, conclui-se que o

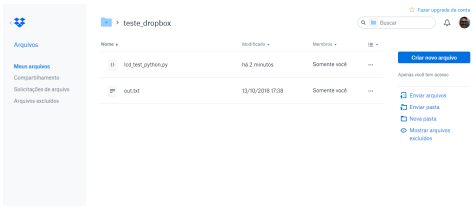

Figura 11. Teste da Conexão RPi - Dropbox no browser

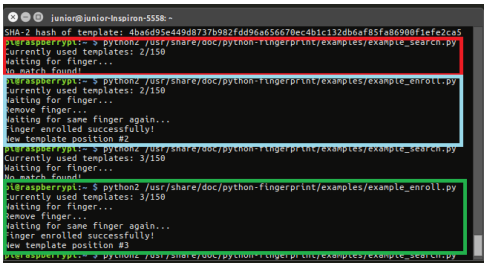

Figura 12. Teste da Conexão RPi - Sensor FPM10A

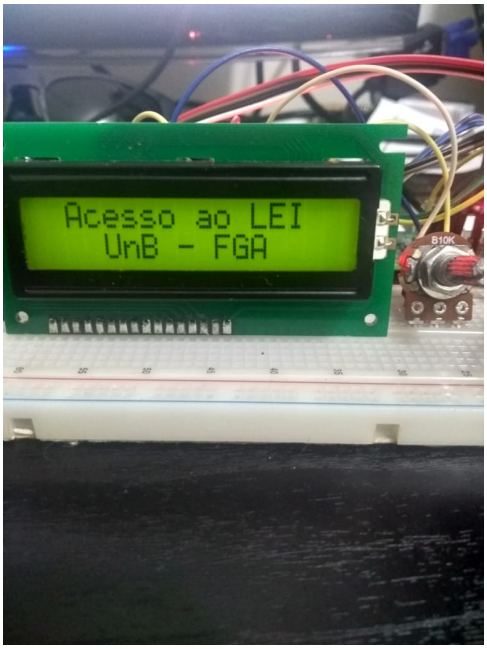

Figura 13. Tela inicial apresentada ao usuário.

mesmo está dimensionado de acordo com o tempo de excussão proposto na disciplina de Sistemas Embarcados.

No segundo ponto de controle foi testada a comunicação do sensor biométrico com a Raspberry Pi além do cadastro e validação de impressão digital. Além disso, foi realizado o teste de display e ambos apresentaram bom desempenho isoladamente.

Para o terceiro ponto de controle foi proposto um *software* para gerenciar o cadastro das digitais e incluir outras informações referentes ao usuário. O código apresentado atingiu o objetivo esperado e ainda incluiu a apresentação de instruções

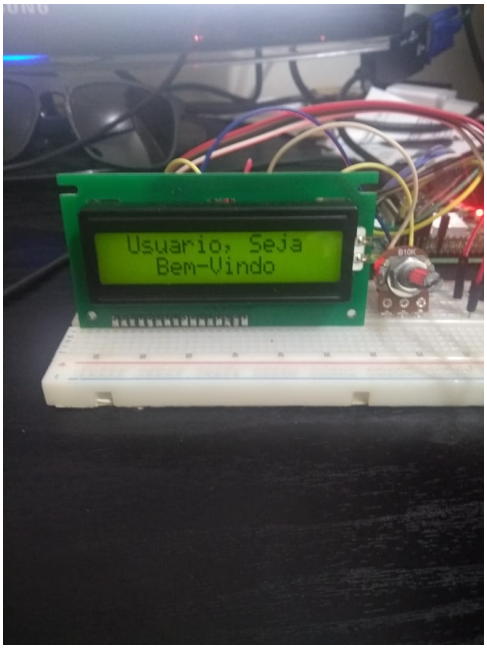

Figura 14. Tela apresentada ao usuário após liberação do acesso.

Figura 15. Tela apresentada ao usuário sem liberação de acesso.

no display LCD.

Para o quarto ponto de controle era esperado que o sistema fosse capaz de cadastrar e excluir usuários, bem como apresentar informações referentes ao acesso em um display para o usuário. Todos esses objetivos foram alcançados com sucesso.

REFERÊNCIAS

[1] FARIA, Diego Resende. Reconhecimento de impressões digitais com baixo custo computacional para um sistema de controle de acesso. 2005.

[2] OLIVIA, Como funciona um sistema de controle de acesso?, 2015. Disponível em: <http://www.graberalarmes.com.br/blog/como-funciona-um-sistema-de-controle-de-acesso/>. Acesso em 04 de set. 2018.

[3] Datasheet LCD16x2 INTECH ITM1602BSTL em: <http://www.intech-lcd.com/image/Character_LCM/1602b-c.pdf>. Acesso em 19 de out. 2018.

[4] Datasheet Raspberry Pi 3 Model B. Disponível em: <https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf>. Acesso em 19 de out. 2018.

[5] Dropbox-Uploader. Disponível em: <https://github.com/andreafabrizi/Dropbox-Uploader>. Acesso em 19 de out. 2018.

[6] Adafruit Python CharLCD. Disponível em: <https://github.com/adafruit/Adafruit_Python_CharLCD>. Acesso em 19 de out. 2018.

[7] Python Fingerprint. Disponível em: <https://github.com/bastianraschke/pyfingerprint>. Acesso em 19 de out. 2018.

APÊNDICE

```
//Codigo do Programa Principal
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <curses.h>
#include <stdio_ext.h>
#include <wiringPi.h>
#include <lcd.h>
#include <time.h>
#include <pthread.h>
#include <unistd.h>
#include <fcntl.h>
#include <signal.h>

//USE WIRINGPI PIN NUMBERS
#define LCD_RS 25 //Register select pin
#define LCD_E 24 //Enable Pin
#define LCD_D4 23 //Data pin 4
#define LCD_D5 22 //Data pin 5
#define LCD_D6 21 //Data pin 6
#define LCD_D7 14 //Data pin 7

//Struct do Usuario
typedef struct
{
      char nome[60];
      char data_aniversario[40];
      char email[50];
      char numero[3];
}dados_usuario;

char etapa;
int contagem = 0, controle = 0;

//Declaracao das Funcoes
char validaOpcao(char opcao);
void novoUsuario();
int deletaUsuario();
void mostraUsuarios();

void *le_digital (void *arg)
{
      int fp;
      fp = open("out.txt", O_WRONLY);
      close(fp);
      if(controle == 0)
      {
            system("rm␣out.txt");
      }
      do
      {
            system("python␣le_digital.py␣|
               ↪ ␣pidof␣python␣>>␣out.txt
               ↪ ");
            system("rm␣out.txt");
      }
      while(1);
      return NULL;
}

void tratamento_alarme(int sig)
{
      char pid[50], c, comando[] = "kill␣"
         ↪ ;
      int fp, i;
      alarm(1);
      i = 0;
      fp = open("out.txt",O_RDONLY);
      if(fp == -1)
            printf("O␣arquivo␣nao␣pode␣ser
               ↪ ␣aberto");
      else
      {

            while((read(fp, &c, 1)) != 0)
            {
                  pid[i] = c;
                  i++;
            }

            close(fp);
            pid[i] = '\0';
            strcat(comando, pid);
            system(comando);

            system("rm␣out.txt");
            pid[0] = '\0';
      }

}
```

```
int main(int argc, char const *argv[])
{
	//Declaracoes
	char opcao;
	FILE *arquivo;
	int lcd, i, fp;
	char pid[10], c, comando[] = "kill " ;

	signal(SIGALRM, tratamento_alarme);

	pthread_t t;


	//Instrucoes

	//Apresenta o Menu
	wiringPiSetup();
	lcd = lcdInit (2, 16, 4, LCD_RS,
	    LCD_E, LCD_D4, LCD_D5, LCD_D6,
	     LCD_D7, 0, 0, 0, 0);

	lcdClear(lcd);

	lcdPosition(lcd, 0, 0);
	lcdPuts(lcd, " Acesso ao LEI");
	lcdPosition(lcd, 0, 1);
	lcdPuts(lcd, "   UnB - FGA");

	system("clear");
	printf("Bem vindo ao Sistema de
	    Controle de Acesso do LEI\n");
	printf("Laboratorio de Engenharia e
	    Inovacao - UnB Gama\n");
	printf("\n");
	printf("\n");
	printf("\n");
	printf("\n");
	printf("\n");
	printf("\n");
	printf("\n");
	system("sleep 2");
	do
	{
		pthread_create(&t, NULL,
		    le_digital, NULL);
		controle++;
		lcdClear(lcd);
		lcdPosition(lcd, 0, 0);
		lcdPuts(lcd, " Acesso ao LEI")
		    ;
		lcdPosition(lcd, 0, 1);
		lcdPuts(lcd, "   UnB - FGA");

		__fpurge(stdin);
		opcao = 0;
		//printf("%c\n", opcao);
		system("clear");
		printf("\t\t\t Menu de Opcoes:
		    \n");
		contagem = 0;
		printf("\n");
		printf("\n");
		printf("\n");
		printf("N - Novo Usuario\n");
		printf("E - Editar Usuario\n")
		    ;
		printf("D - Deletar Usuario\n"
		    );
		printf("S - Sair do Menu\n");
		printf("\n");
		printf("\n");
		printf("\n");
		//system("python3 opcao_menu.
		    py");
		//arquivo = fopen("pipe.txt",
		    "r");
		//opcao = fgetc(arquivo);
		//fclose(arquivo);
		//system("rm pipe.txt");
		if(!contagem)
		{
			printf("Digite a opcao
			    desejada: ");
			opcao = validaOpcao(
			    fgetc(stdin));

		}
		printf("\n\n\n\n");
		contagem = 1;
		__fpurge(stdin);
		//clean_buffer();
		switch(opcao)
		{
			case 'N':
			{
				system("./
				    kill_teste")
				    ;
				pthread_cancel(t);
				//pthread_join(t,
				    NULL);
				novoUsuario();
				break;
			}
			case 'D':
			{
				system("./
				    kill_teste")
				    ;
				pthread_cancel(t);
				deletaUsuario();
				break;
			}
```

```
                case 'M':
                {
                      system("./
                         ↪ kill_teste")
                         ↪ ;
                      pthread_cancel(t);
                      mostraUsuarios();
                      break;
                }
          }
      }while(opcao != 'S');
      lcdClear(lcd);
      system("./kill_teste");
      pthread_cancel(t);
      return 0;
}

void novoUsuario()
{
      //Declaracao de variaveis
      FILE *arquivo;
      int fp;
      dados_usuario temp;
      char string[] = "mkdir ", temporaria
         ↪ [50], temporaria2[50];
      int i, j = 0;
      char c, nome[50];
      int lcd;
      signal(SIGALRM, tratamento_alarme);

      wiringPiSetup();
   lcd = lcdInit (2, 16, 4, LCD_RS, LCD_E,
      ↪  LCD_D4, LCD_D5, LCD_D6, LCD_D7,
      ↪ 0, 0, 0, 0);

      lcdClear(lcd);

      lcdPosition(lcd, 0, 0);
      lcdPuts(lcd, "  Cadastro de");
      lcdPosition(lcd, 0, 1);
      lcdPuts(lcd, "    Usuarios");


      //Menu de apresentacao
      __fpurge(stdin);
      __fpurge(stdout);
      system("clear");
      printf("Cadastro de Usuario - 
         ↪ Laboratorio Lei - UnB Gama\n")
         ↪ ;
      printf("\n");
      printf("\n");
      printf("\n");

      //Digitar o nome do usuario
      printf("\n\n\nDigite o nome no 
         ↪ formato\n");
      printf("NOME SOBRENOME: ");
      fgets(temp.nome, 60*sizeof(char),
         ↪ stdin);
      if(temp.nome[strlen(temp.nome)-1] ==
         ↪  '\n')
            temp.nome[strlen(temp.nome)-1]
               ↪  = '\0';
      __fpurge(stdin);
      __fpurge(stdout);

      //Digitar o email do usuario
      printf("\n\n\nDigite o email: ");
      fgets(temp.email, 50*sizeof(char),
         ↪ stdin);
      __fpurge(stdin);
      __fpurge(stdout);

      //Digitar a data de nascimento do
         ↪ usuario
      printf("\n\n\nDigite a data de 
         ↪ nascimento\n");
      printf("no formato DD/MM/AAAA: ");
      fgets(temp.data_aniversario, 40*
         ↪ sizeof(char), stdin);
      __fpurge(stdin);
      __fpurge(stdout);


      /*
      system("python3 move_python.py");
      system("./upload.sh");
      */
      system("python nova_digital.py");

      fp = open("position.txt", O_RDONLY);
      //fp = fopen("position.txt", "r");
      temp.numero[0] = '\0';
      i = 0;
      while(read(fp, &c, 1) != 0)
      {
            temp.numero[i] = c;
            i++;
      }
      temp.numero[i] = '\0';

      close(fp);
      system("rm position.txt");


      //Salva os dados do usuario no
         ↪ arquivo geral ".bin"
      if((arquivo = fopen("dados_usuario.
         ↪ bin", "ab")) == NULL)
      {
            printf("O arquivo nao pode ser
               ↪  aberto.");
            fclose(arquivo);
      }
      else
```

```
      fwrite(&temp, sizeof(temp), 1,
         ↪ arquivo);
fclose(arquivo);

//Cria a pasta do usuario
strcpy(temporaria, temp.nome);


//strncpy(temporaria, temporaria,
   ↪ strlen(temporaria)-1);


for(i = 0; temporaria[i]; i++)
{
      if(temporaria[i] == ' ')
            temporaria[i] = '_';
}
strcat(string, temporaria);
system(string);

//Salva os dados do usuario no
   ↪ arquivo ".txt"
strcpy(temporaria2, temporaria);
strcpy(temp.nome, temporaria2);


strcat(temporaria2, ".txt");
arquivo = fopen(temporaria2, "w");
if(!arquivo)
{
      printf("O arquivo nao pode ser
         ↪ aberto");
      exit(-1);
}
for(i = 0; temp.nome[i]; i++)
      fputc(temp.nome[i], arquivo);
fputc('\n', arquivo);

for(i = 0; temp.email[i]; i++)
      fputc(temp.email[i], arquivo);

for(i = 0; temp.data_aniversario[i];
   ↪ i++)
      fputc(temp.data_aniversario[i
         ↪ ], arquivo);

for(i = 0; temp.numero[i]; i++)
      fputc(temp.numero[i], arquivo)
         ↪ ;

fclose(arquivo);



//Salva o arquivo de dados do
   ↪ usuario na pasta do usuario
strcpy(string, "mv ");
strcat(string, temporaria2);
strcat(string, " ");
strcat(string, temporaria);
system(string);

//system("move.sh");
strcpy(string, "mv /tmp/fingerprint.
   ↪ bmp ~/pc3_embarcados/");
strcat(string, temporaria);

system(string);

temp.nome[0] = '\0';
temp.email[0] = '\0';
temp.data_aniversario[0] = '\0';
temp.numero[0] = '\0';
}

int deletaUsuario()
{
      //Declaracoes
      FILE *arquivo, *temp_file;
      char nome[60], confirma,
         ↪ nome_usuario[60], comando[] =
         ↪ "rm -r ", deletar[] = "echo ";
      int posicao, position_cpy, i;
      dados_usuario temp, usuario;

      //Instrucoes
      signal(SIGALRM, tratamento_alarme);

      //Pede o nome do usuario
      system("clear");
      printf("Digite o nome do usuario que
         ↪ sera apagado: ");
      fgets(nome, sizeof(nome), stdin);
      __fpurge(stdin);
      if(nome[strlen(nome)-1] == '\n')
            nome[strlen(nome)-1] = '\0';

      puts(nome);



      strcpy(nome_usuario, nome);

      for(i = 0; nome[i]; i++)
      {
            if(nome_usuario[i] == ' ')
                  nome_usuario[i] = '_';
      }

      //Abre o arquivo
      arquivo = fopen("dados_usuario.bin",
         ↪ "rb");
      if(!arquivo)
            puts("O arquivo nao pode ser
               ↪ aberto");
      else
```

```
{
        rewind(arquivo);
        posicao = ftell(arquivo);
        while(!feof(arquivo) &&
            ↪ posicao == 0)
        {
                //Se o usuario existir,
                    ↪ encontramos a
                    ↪ posicao dele no
                    ↪ arquivo.
                fread(&temp, sizeof(
                    ↪ dados_usuario), 1,
                    ↪  arquivo);
                if((strcmp(nome, temp.
                    ↪ nome)) == 0)
                        posicao = ftell(
                            ↪ arquivo);
        }
        fclose(arquivo);
        if(!posicao)
        {
                printf("nao␣deveria␣
                    ↪ entrar␣aqui␣se␣%d␣
                    ↪ nao␣e␣igual␣a␣zero
                    ↪ ", posicao);
                //Senao, sai fora!
                printf("O␣usuario␣nao␣
                    ↪ existe\n\n\n");
                system("sleep␣2");
                system("clear");
                return 0;
        }
        else
        {
                arquivo = fopen("
                    ↪ dados_usuario.bin"
                    ↪ , "rb");
                rewind(arquivo);
                //Administrador verifica
                    ↪  se este usuario e
                    ↪  o que deve ser
                    ↪ excluido
                printf("\n\n\n\n");
                system("clear");
                fseek(arquivo, posicao-
                    ↪ sizeof(
                    ↪ dados_usuario),
                    ↪ SEEK_SET);
                fread(&usuario, sizeof(
                    ↪ dados_usuario), 1,
                    ↪  arquivo);
                fclose(arquivo);

                printf("Nome:␣%s\n",
                    ↪ usuario.nome);
                printf("E-mail:␣%s\n",
                    ↪ usuario.email);
                printf("Data␣de␣
                    ↪ Nascimento:␣%s\n",
                    ↪  usuario.
                    ↪ data_aniversario);
                printf("Numero:␣%s\n",
                    ↪ usuario.numero);
                printf("\n\n");
                printf("Deseja␣mesmo␣
                    ↪ excluir␣este␣
                    ↪ usuario?\n");
                printf("Digite␣'S'␣para␣
                    ↪ sim␣e␣'N'␣para␣nao
                    ↪ .");

                scanf("%c", &confirma);
                __fpurge(stdin);


                if(toupper(confirma) ==
                    ↪ 'S')
                {
                        temp_file = fopen(
                            ↪ "temp.bin",
                            ↪ "wb");
                        fclose(temp_file);
                        arquivo = fopen("
                            ↪ dados_usuario
                            ↪ .bin", "rb")
                            ↪ ;
                        //Se sim, copiamos
                            ↪  todo o
                            ↪ arquivo,
                            ↪ exceto o
                            ↪ usuario a
                            ↪ ser excluido
                        while(!feof(
                            ↪ arquivo))
                        {
                                fread(&temp,
                                    ↪
                                    ↪ sizeof
                                    ↪ (
                                    ↪ dados_usuario
                                    ↪ ), 1,
                                    ↪ arquivo
                                    ↪ );
                                if(strcmp(
                                    ↪ nome,
                                    ↪ temp.
                                    ↪ nome)
                                    ↪ != 0)
                                {
                                        temp_file
                                            ↪
                                            ↪ =
                                            ↪
                                            ↪ fopen
                                            ↪ (
                                            ↪ "
```

```
    ↪ temp
    ↪ .
    ↪ bin
    ↪ "
    ↪ ,
    ↪
    ↪ "
    ↪ a
    ↪ +
    ↪ b
    ↪ "
    ↪ )
    ↪ ;
    ↪
if(!
    ↪ temp_file
    ↪ )
    ↪
{
       printf
           ↪ (
           ↪ "
           ↪ O
           ↪ ␣
           ↪ arquivo
           ↪ ␣
           ↪ nao
           ↪ ␣
           ↪ pode
           ↪ ␣
           ↪ ser
           ↪ ␣
           ↪ aberto
           ↪ "
           ↪ )
           ↪ ;
           ↪
       exit
           ↪ (1)
           ↪ ;
           ↪
}
fwrite
    ↪ (&
    ↪ temp
    ↪ ,
    ↪
    ↪ sizeof
    ↪ (
    ↪ temp
    ↪ )
    ↪ ,
    ↪
    ↪ 1,
    ↪
    ↪ temp_file
    ↪ )
    ↪ ;
                ↪
            fclose
                ↪ (
                ↪ temp_fi
                ↪ )
                ↪ ;
                ↪
      }
}
fclose(arquivo);
//Remove o arquivo
    ↪  atual
remove("
    ↪ dados_usuario
    ↪ .bin");

//Renoemia o
    ↪ arquivo
    ↪ temporario
rename("temp.bin",
    ↪  "
    ↪ dados_usuario
    ↪ .bin");

//Deleta a pasta
    ↪ do usuario
    ↪ do diretorio
    ↪  do projeto
strcpy(
    ↪ nome_usuario
    ↪ , usuario.
    ↪ nome);
for(i = 0;i <
    ↪ strlen(
    ↪ nome_usuario
    ↪ )-1; i++)
{
       if(
           ↪ nome_usuario
           ↪ [i] ==
           ↪  '␣')
              nome_usuario
                  ↪ [
                  ↪ i
                  ↪ ]
                  ↪
                  ↪ =
                  ↪
                  ↪ '
                  ↪ _
                  ↪ '
                  ↪ ;
                  ↪
}
strcat(comando,
    ↪ nome_usuario
    ↪ );
system(comando);
```

```
arquivo = fopen("
    ↪ dados_usuario
    ↪ .bin", "rb")
    ↪ ;
fseek(arquivo,
    ↪ posicao -
    ↪ sizeof(
    ↪ dados_usuario
    ↪ ), SEEK_SET)
    ↪ ;
fread(&temp,
    ↪ sizeof(
    ↪ dados_usuario
    ↪ ), 1,
    ↪ arquivo);
fclose(arquivo);

/*
arquivo = fopen("
    ↪ pipe_delete.
    ↪ txt", "w");
for(i = 0; temp.
    ↪ numero[i]; i
    ↪ ++)
      if(temp.
          ↪ numero
          ↪ [i] >
          ↪ '0' &&
          ↪  temp.
          ↪ numero
          ↪ [i] <
          ↪ '9')
      {
            fputc(
                ↪ temp
                ↪ .
                ↪ numero
                ↪ [
                ↪ i
                ↪ ],
                ↪
                ↪ arquivo
                ↪ )
                ↪ ;
                ↪
      }
fclose(arquivo);
*/
puts(temp.numero);
if(atoi(temp.
    ↪ numero) < 0)
{
      system("echo
          ↪ ␣0␣>>␣
          ↪ pipe_delete
          ↪ .txt")
          ↪ ;
}
else{
      strcat(
          ↪ deletar
          ↪ , temp
          ↪ .
          ↪ numero
          ↪ );
      strcat(
          ↪ deletar
          ↪ , "␣>>
          ↪ ␣
          ↪ pipe_delete
          ↪ .txt")
          ↪ ;
}
system(deletar);

system("python␣
    ↪ deleta_digital
    ↪ .py");
system("rm␣
    ↪ pipe_delete.
    ↪ txt");

system("clear");
printf("O␣usuario␣
    ↪ %s␣foi␣
    ↪ deletado␣com
    ↪ ␣sucesso.\n\
    ↪ n\n",
    ↪ usuario.nome
    ↪ );
system("sleep␣2");
system("clear");

temp.nome[0] = '\0
    ↪ ';
temp.email[0] = '
    ↪ \0';
temp.
    ↪ data_aniversario
    ↪ [0] = '\0';
temp.numero[0] = '
    ↪ \0';

usuario.nome[0] =
    ↪ '\0';
usuario.email[0] =
    ↪  '\0';
usuario.
    ↪ data_aniversario
    ↪ [0] = '\0';
usuario.numero[0]
    ↪ = '\0';

return 0;
```

```
                }else
                {
                        //Senao, sai fora!
                        fclose(arquivo);
                        return 0;
                }
            }
        }
}

void mostraUsuarios()
{
        FILE *fp;
        char c;
        dados_usuario temp;


        fp = fopen("dados_usuario.bin", "rb"
            ↪ );
        while(!feof(fp))
        {
                if(!feof(fp))
                {
                        fread(&temp, sizeof(
                            ↪ dados_usuario), 1,
                            ↪  fp);
                        puts(temp.nome);
                        puts(temp.email);
                        puts(temp.
                            ↪ data_aniversario);
                        puts(temp.numero);
                }
        }
        fclose(fp);

        scanf("%c", &c);

}

char validaOpcao(char opcao)
{
        signal(SIGALRM, tratamento_alarme);
        while(opcao != 'M' && opcao != 'N'
            ↪ && opcao != 'D' && opcao != 'S
            ↪ ')
        {
                printf("Opcao␣invalida,␣digite
                    ↪ ␣novamente:␣");
                opcao = fgetc(stdin);
                __fpurge(stdin);
        }

}

#Codigo da Leitura da Digital - le_digital
    ↪ .py

#!/usr/bin/env python

# -*- coding: utf-8 -*-

"""
PyFingerprint
Copyright (C) 2015 Bastian Raschke <
    ↪ bastian.raschke@posteo.de>
All rights reserved.
"""
from Adafruit_CharLCD import
    ↪ Adafruit_CharLCD
import time
import tempfile
import os
import sys
import hashlib
from pyfingerprint.pyfingerprint import
    ↪ PyFingerprint

##Inicializacao do LCD


lcd = Adafruit_CharLCD(rs = 26, en = 19,
    ↪ d4 = 13, d5 = 6, d6 = 5, d7 = 11,
    ↪ cols = 16, lines = 2)

lcd.message("␣Acesso␣ao␣LEI\n")
lcd.message("␣␣␣UnB␣-␣FGA")

sys.stdout.flush()
sys.stdout.close()

sys.stderr.flush()
sys.stderr.close()
## Search for a finger
##

## Tries to initialize the sensor
try:
    f = PyFingerprint('/dev/ttyUSB0',
        ↪ 57600, 0xFFFFFFFF, 0x00000000)

    if ( f.verifyPassword() == False ):
        raise ValueError('The␣given␣
            ↪ fingerprint␣sensor␣password␣is
            ↪ ␣wrong!')

except Exception as e:
    print('The␣fingerprint␣sensor␣could␣not
        ↪ ␣be␣initialized!')
    print('Exception␣message:␣' + str(e))
    exit(1)



## Gets some sensor information
#print('Currently used templates: ' + str(
    ↪ f.getTemplateCount()) +'/'+ str(f.
    ↪ getStorageCapacity()))
```

```
## Tries to search the finger and
   ↪ calculate hash
try:
   #print('Waiting for finger...')

   ## Wait that finger is read
   while ( f.readImage() == False ):
      pass



   ## Converts read image to
      ↪ characteristics and stores it in
      ↪ charbuffer 1
   f.convertImage(0x01)

   ## Searchs template
   result = f.searchTemplate()

   positionNumber = result[0]
   accuracyScore = result[1]

   if ( positionNumber == -1 ):
      lcd.clear()
      lcd.message("   Usuario\n")
      lcd.message(" nao cadastrado")
      os.system("sleep 6")
      lcd.clear()
      exit(0)
   else:
      lcd.clear()
      lcd.message(" Usuario, Seja\n")
      lcd.message("   Bem-Vindo")
      os.system("sleep 6")
      lcd.clear()
    # print('Found template at position #'
      ↪  + str(positionNumber))
    # print('The accuracy score is: ' +
      ↪ str(accuracyScore))
   a = 0
   ## OPTIONAL stuff
   ##

   ## Loads the found template to
      ↪ charbuffer 1
   #f.loadTemplate(positionNumber, 0x01)

   ## Downloads the characteristics of
      ↪ template loaded in charbuffer 1
   #characterics = str(f.
      ↪ downloadCharacteristics(0x01)).
      ↪ encode('utf-8')

   ## Hashes characteristics of template
   #print('SHA-2 hash of template: ' +
      ↪ hashlib.sha256(characterics).
      ↪ hexdigest())
```

```
except Exception as e:
   print('Operation failed!')
   print('Exception message: ' + str(e))
   exit(1)

#Codigo do Cadastro da Digital

from Adafruit_CharLCD import
   ↪ Adafruit_CharLCD
import time
import tempfile
import os
from pyfingerprint.pyfingerprint import
   ↪ PyFingerprint

##Inicializacao do LCD


lcd = Adafruit_CharLCD(rs = 26, en = 19,
   ↪ d4 = 13, d5 = 6, d6 = 5, d7 = 11,
   ↪ cols = 16, lines = 2)
## Enrolls new finger
##

## Tries to initialize the sensor
try:
   f = PyFingerprint('/dev/ttyUSB0',
      ↪ 57600, 0xFFFFFFFF, 0x00000000)

   if ( f.verifyPassword() == False ):
      raise ValueError('The given
         ↪ fingerprint sensor password is
         ↪  wrong!')


except Exception as e:
   print('O sensor de impressao digital
      ↪ nao pode ser inicializado')
   print('Exception message: ' + str(e))
   exit(1)

## Gets some sensor information
#print('Templates Usados: ' + str(f.
   ↪ getTemplateCount()) +'/'+ str(f.
   ↪ getStorageCapacity()))

## Tries to enroll new finger
try:
   print('Coloque o dedo no sensor ...')
   lcd.clear()
   lcd.message("Coloque o dedo \n")
   lcd.message("no sensor (1/3)")

   ## Wait that finger is read
   while ( f.readImage() == False ):
      pass
```

```
   ## Converts read image to
      ↪ characteristics and stores it in
      ↪ charbuffer 1
   f.convertImage(0x01)

   ## Checks if finger is already enrolled
   result = f.searchTemplate()
   positionNumber = result[0]

   if ( positionNumber >= 0 ):
      print('A digital ja esta cadastrada
         ↪ na posicao #' + str(
         ↪ positionNumber))
      exit(1)

   print('Retire o dedo do sensor ...')
   lcd.clear()
   lcd.message("Retire o dedo\n")
   lcd.message("do sensor 1/3")
   time.sleep(2)

   print('Coloque o mesmo dedo no sensor
      ↪ ...')
   lcd.clear()
   lcd.message("Coloque o dedo\n")
   lcd.message("no sensor 2/3")


   ## Wait that finger is read again
   while ( f.readImage() == False ):
      pass

   ## Converts read image to
      ↪ characteristics and stores it in
      ↪ charbuffer 2
   f.convertImage(0x02)

   ## Compares the charbuffers
   if ( f.compareCharacteristics() == 0 ):
      raise Exception('Dedos nao sao os
         ↪ mesmos.')
   print('Retire o dedo do sensor ...')
   lcd.clear()
   lcd.message("Retire o dedo\n")
   lcd.message("do sensor 2/3")
   time.sleep(2)

   ## Creates a template
   f.createTemplate()

   ## Saves template at new position
      ↪ number
   positionNumber = f.storeTemplate()
################################################
   ↪
################## Download da imagem
   ↪ ##############################
################################################
      ↪
      print('Coloque o mesmo dedo no sensor
         ↪ mais uma vez...')
      lcd.clear()
      lcd.message("Coloque o dedo\n")
      lcd.message("no sensor 3/3")

   ## Wait that finger is read
      while ( f.readImage() == False ):
         pass
      #lcd.clear()
      #lcd.message("Retire o dedo\n")
      #lcd.message("do sensor 3/3")
      print("Retire o dedo do sensor ...")
      lcd.clear()
      lcd.message("Retire o dedo\n")
      lcd.message("do sensor 3/3")
      time.sleep(2)

      print('Aguarde o final da operacao...')
      lcd.clear()
      lcd.message("Aguarde ...")

      imageDestination = tempfile.gettempdir
         ↪ () + '/fingerprint.bmp'
      f.downloadImage(imageDestination)

      #print('The image was saved to "' +
         ↪ imageDestination + '".')

   ################################################
      ↪

      print('Digital cadastrada com sucesso
         ↪ .')
      print('Nova digital cadastrada na
         ↪ posicao #' + str(positionNumber))
      lcd.clear()
      lcd.message("Digital Salva\n")
      lcd.message("na posicao " + str(
         ↪ positionNumber))

      f = open("position.txt", "w")
      f.write(str(positionNumber))
      f.close()

   except Exception as e:
      print('Operation failed!')
      print('Exception message: ' + str(e))
      exit(1)

   #Codigo para deletar a digital
#########################################################
   ↪
################## Download da imagem
   ↪ ##############################
################################################
```

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

"""
Python interface
```

```
Copyright (C) 2015 Bastian Raschke <
    ↪ bastian.raschke@posteo.de>
All rights reserved.
"""

from pyfingerprint.pyfingerprint import
    ↪ PyFingerprint


## Deletes a finger from sensor
##


## Tries to initialize the sensor
try:
    f = PyFingerprint('/dev/ttyUSB0',
        ↪ 57600, 0xFFFFFFFF, 0x00000000)

    if ( f.verifyPassword() == False ):
        raise ValueError('The given
            ↪ fingerprint sensor password is
            ↪  wrong!')

except Exception as e:
    print('The fingerprint sensor could not
        ↪  be initialized!')
    print('Exception message: ' + str(e))
    exit(1)

## Gets some sensor information
print('Currently used templates: ' + str(f
    ↪ .getTemplateCount()) +'/'+ str(f.
    ↪ getStorageCapacity()))

## Tries to delete the template of the
    ↪ finger
try:
        #positionNumber = input('Please
            ↪ enter the template position
            ↪ you want to delete: ')
        #positionNumber = int(positionNumber
            ↪ )

    fd = open("pipe_delete.txt", "r")
    positionNumber = int(fd.read())
    print(positionNumber)
    fd.close()

    if (f.deleteTemplate(positionNumber) ==
        ↪  True):
        print('Template deleted!')
    else:
        print("nao foi possivel deletar o
            ↪ usuario")

except Exception as e:
    print('Operation failed!')
    print('Exception message: ' + str(e))
                exit(1)
```