

Dispositivo para gerenciamento de comprimidos

Mateus Alves da Rocha

Engenharia Eletrônica

Universidade de Brasília-UnB/FGA Gama, DF.

e-mail: mateus.alves.unb@gmail.com

Mayara Barbosa dos Santos

Engenharia Eletrônica

Universidade de Brasília-UnB/FGA Gama, DF.

e-mail: mayara.b97@gmail.com

Abstract—Com o advento dos vários problemas de saúde, as pessoas vêm-se obrigadas a tomar vários remédios. Com o intuito de ajudar os usuários para que não esqueçam de ingerir seus medicamentos e acarretarem problemas mais sérios por falta de seu uso, este projeto tem o propósito de auxiliá-los no dia-a-dia e um dispositivo para gerenciamento de comprimidos será desenvolvido por intermédio do microcontrolador *Raspberry Pi*.

Palavras-chave - Gerenciamento, comprimidos, *Raspberry Pi*

I. INTRODUÇÃO

Sabe-se que, com o envelhecimento, há uma tendência de diminuição na capacidade de memorização de um indivíduo. Isto é agravado, caso esta pessoa sofra de alguma doença que afete diretamente nessa habilidade.[1]

Entretanto, não é incomum que pessoas esqueçam coisas importantes independente de enfermidades relacionadas à perda de memória ou o envelhecimento. O desenvolvimento tecnológico pode ter desempenhado um papel neste problema, dado que desde que os *smartphones* facilitaram o uso de agendas para contatos é fácil encontrar alguém que não tenha memorizado o próprio número de sua residência, por exemplo. Apesar disso, o esquecimento de contatos ou termos que podem ser facilmente encontrados com poucos minutos de pesquisa na internet não constitui um problema sério para a população. Porém, um dos problemas da falta de exercício no sentido de melhorar a capacidade de memorização é a dificuldade em seguir prescrições médicas de medicamentos.

Os idosos são os mais afetados por este problema. Uma pesquisa realizada na Universidade Estadual de Campinas e publicada na revista *Ciências e Saúde Coletiva* entrevistou 165 idosos e constatou que 58,2% possui acima de quatro comorbidades simultâneas o que leva a um número considerável de remédio para gerenciar ao longo do dia.[2] E, neste mesmo estudo, 55,2% afirmou não ter cuidador. A necessidade de vários comprimidos diariamente e em horários diferentes é dificultado pelo esquecimento, trabalho e déficit cognitivo.[3]

No entanto, como mencionado anteriormente, não são apenas idosos que possuem dificuldades relacionadas ao número simultâneo de medicamentos. Kourrouski e Lima publicaram um estudo na revista *Latino Americana de*

Enfermagem que apesar de os adolescentes diagnosticados como portadores do HIV relatarem saber dos benefícios da medicação no controle uma grande parcela deles não adere aos tratamentos por diversos fatores e dentre eles inclui-se o esquecimento do medicamento. Essas pesquisadoras afirmam ainda que é necessário orientá-los para uso de despertadores para que não esqueçam o horário correto das medicações. [4]

Há sistemas comerciais desenvolvidos voltados ao gerenciamento de medicamentos. Como os sistemas da 1. Entretanto, os preços dessas tecnologias ainda estão pouco acessíveis a grande parte da população. O produto *MedFolio Wireless Pillbox* modelo WP1050 custa \$250,95 de acordo com o site da *Amazon* [5]. Já o *MedMinder Maya* funciona a partir de assinatura que variam de \$40 a \$60 por mês. [6]



Fig. 1. Sistema de gerenciamento de medicamentos já desenvolvido no mercado.

Existem também alguns sistemas amadores que buscam atender essa demanda. Um exemplo é o dispositivo de Wojtek Siudzinski que utiliza um servo motores para fazer a movimentação de discos impressos em uma impressora 3D para dispensar pílulas de MM, mas pode ser utilizado também para comprimidos. [7]

Thomas Nabelek e Adam Nolte, alunos da Universidade de Missouri nos Estados Unidos desenvolveram um projeto que busca automatizar o gerenciamento de comprimidos. O sistema prevê o controle inclusive de farmacêuticos através da possibilidade de acompanhamento da rotina de remédios através da Web. [8]

Neste cenário, o sistema desenvolvido neste trabalho traz uma solução tecnológica para o controle de medicamentos. Busca-se retirar dos pacientes a responsabilidade desse

gerenciamento e ao mesmo tempo garantir uma alta confiabilidade que os medicamentos serão lembrados e administrados conforme prescritos pelos profissionais da saúde.

II. OBJETIVOS

O objetivo do projeto é desenvolver um sistema que auxilie o usuário ingerir seus comprimidos corretamente de forma que facilite sua rotina e não interrompa seu tratamento. O produto em questão terá um banco de dados com todos os usuários que serão cadastrados assim como todas as informações pertinentes para que o usuário insira o remédio adequadamente, o usuário terá fácil acesso ao sistema por meio de reconhecimento facial.

III. REQUISITOS

Utilizando o hardware *Raspberry Pi* que comporta diversas distribuições Linux como plataforma de desenvolvimento do produto proposto, o projeto tem os seguintes requisitos:

- Permitir ao administrador do sistema configurar a rotina de horários e quantidade de remédios a serem prescritos. Além de ser possível cadastrar os usuários do sistema que será armazenado em um banco de dados.
- Emitir no horário configurado um aviso por meio de um *buzzer* para o usuário do sistema tomar o remédio na hora certa;
- Identificar o usuário e associar a ele a rotina de administração de remédios específica por meio de reconhecimento facial a partir de uma câmera que fará a comunicação com a *Raspberry Pi*;
- Apresentar um dispositivo eletromecânico para dispensar o remédio automaticamente. O mesmo vai operar com motor de passo e um servo motor controlado pela *Raspberry Pi*;
- O usuário responsável pelo sistema terá que inserir os comprimidos no dispositivo, assim que cada *slot* estiver desocupado.

IV. BENEFÍCIOS

O projeto apresenta os seguintes benefícios:

- 1) O usuário será lembrado da hora que terá que ingerir o remédio;
- 2) O tratamento da doença a ser tratada não será interrompido;
- 3) Evitar problemas mais sérios nos casos de doenças crônicas;
- 4) Baixo custo.

V. HARDWARE

Para o devido funcionamento do projeto, o dispositivo eletromecânico contará com os seguintes componentes:

Tabela 1: Materiais utilizados na confecção do protótipo

Materiais	Quantidade (und)
Raspberry Pi 3	1
Servo motor	2
Motor de passo	1
Led's	3
Estrutura feita na impressora 3	1
Webcam	1
Monitor	1
Buzzer	1

- 1) **Raspberry Pi:** A Raspberry Pi 3 é um computador de baixo custo e portátil, ela suporta o sistema operacional Ubuntu, Raspbian e outras distribuições do Linux. Além disso, é compatível com o Windows 10 IoT, versão do software da Microsoft feita para automação doméstica e outras aplicações envolvendo Internet das Coisas.

O Raspberry Pi 3 pode ser encontrado no mercado com o valor em média de R\$ 200,00. Como na Fig. 2 a Raspberry Pi 3 suporta vários periféricos como quatro portas USB, uma porta HDMI, *WiFi* para conexão com a internet, *slot* para microSD e porta *ethernet*, aumentando suas aplicações com o uso da Raspberry Pi 3.



Fig. 2. Raspberry Pi 3

- 2) **Motor de passo e servomotor:** Um motor de passo como na Fig.3 e Fig. 4 é um recurso eletromecânico onde precisa de um movimento controlado. A rotação do motor é definida com base nos pulsos elétricos gerados e a velocidade do motor de passo é definida pela frequência com que esses pulsos são enviados. O motor e o servo motor servirão para o acionamento da movimentação dos *slots*.

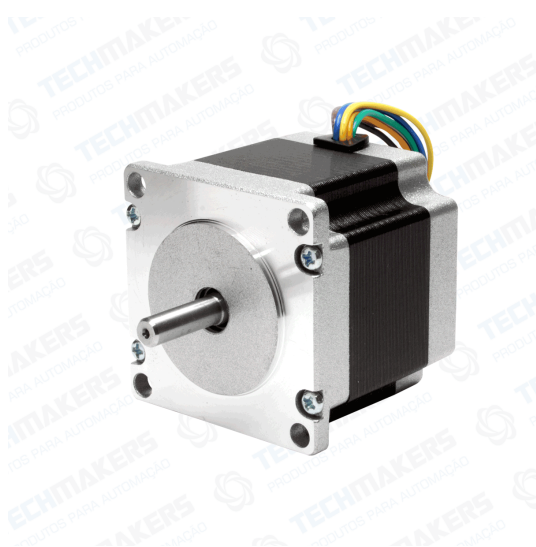


Fig. 3. Motor de passo NEMA.



Fig. 5. Buzzer.

A conexão destes elementos com a *Raspberry Pi* foi feita utilizando os pinos de GPIO e fazendo uso de *jumpers*. A numeração destes pinos obedeceu a estipulada na biblioteca *WiringPi* e pode ser verificada nos códigos em anexo a este documento. Pode-se visualizar estas conexões na Fig. 6 abaixo. Junto a estes componentes, há uma *webcam* conectada na porta USB da placa.

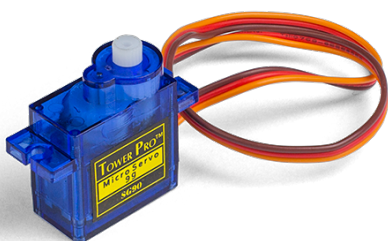


Fig. 4. Servo motor.

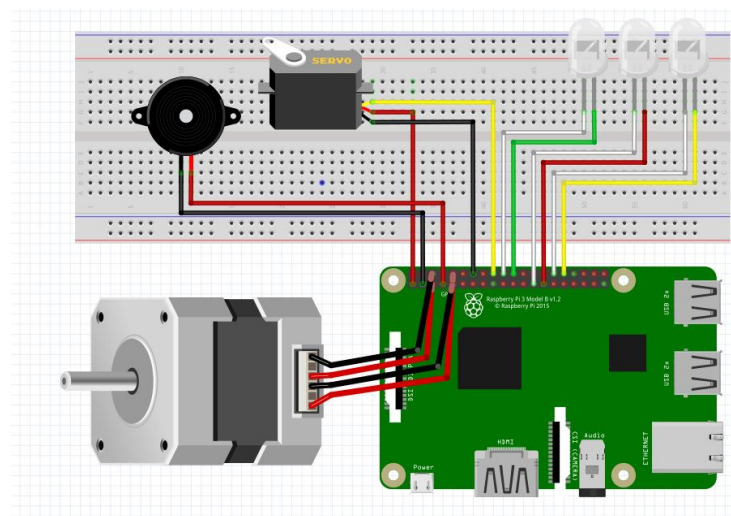


Fig. 6. Conexões componentes periféricos do sistema.

3) Buzzer:

Buzzer é um componente eletrônico da Fig. 5 que recebe uma fonte de energia e através dela emite uma frequência sonora, com isso avisará aos usuários do sistema que está na hora de ingerir o remédio.

VI. SOFTWARE

Os programas utilizados para movimentação do equipamento estão disponíveis no apêndice deste documento. O funcionamento será da seguinte forma: O usuário deverá utilizar o programa de cadastramento previamente para gravar as informações do paciente no banco de dados. Após isso, será gerado um arquivo para armazenar as informações.

Com o banco de dados Fig. 7 preenchido, o programa que estipula os alarmes dos remédio também é o responsável

por recolher as informações no banco de dados e de chamar os códigos dos periféricos necessários para dispensar o medicamento no momento correto. Por exemplo, ao identificar um horário de medicação, o programa lê as informações no banco de dados com o nome e o *slot* onde está localizado o comprimido.

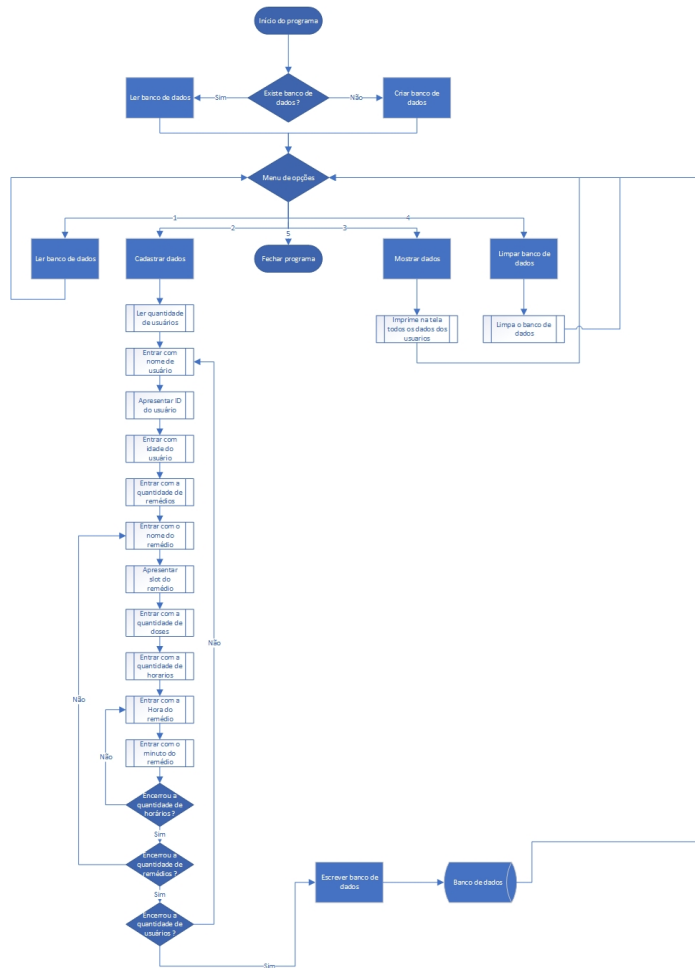


Fig. 7. Fluxograma do banco de dados.

Em seguida, aciona o motor de passo para o giro do eixo principal do programa de forma a posicionar o comprimido em cima da comporta. Após isso é emitido o aviso sonoro Fig. 8 ao usuário indicando a necessidade de se aproximar do aparelho. Quando o usuário se aproxima do dispositivo, sua imagem é capturada pela *Webcam* acoplada ao sistema e a partir deste ponto é realizado um processamento de imagem para identificar se a pessoa certa receberá os comprimidos. Uma vez identificada conforme a Fig. 16, cessam os avisos sonoros e o servo motor é acionado liberando os dispositivos.

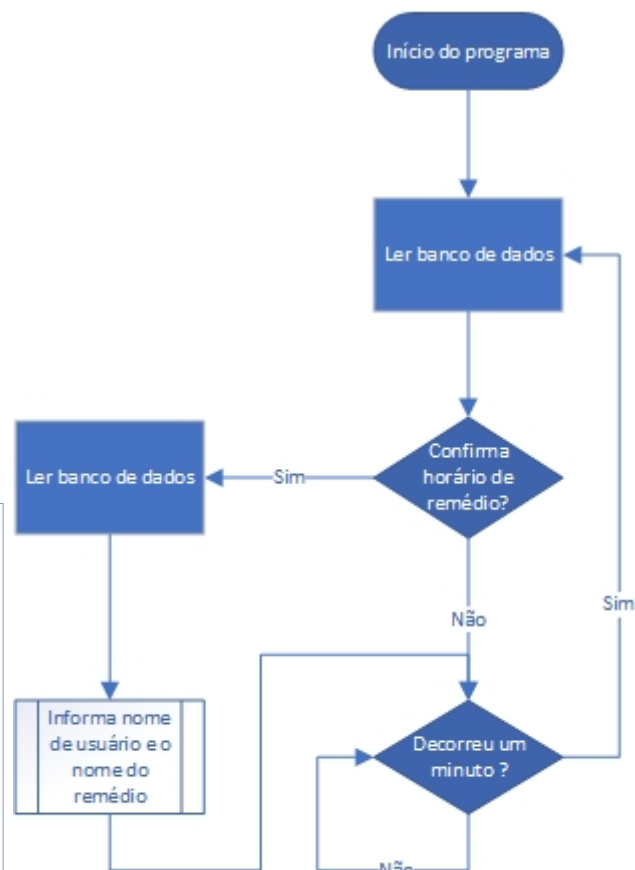


Fig. 8. Fluxograma da hora do acionamento do Buzzer.

Há também algumas indicações luminosas para facilitar o entendimento do processo. Ao iniciar o aviso sonoro uma luz amarela piscará de forma intermitente. Ao iniciar a identificação visual do usuário a luz manterá o brilho constante. Caso seja liberado o comprimido para o usuário, um LED verde acenderá. Caso contrário, um LED vermelho irá acender.

O acionamento dos componentes periféricos é feito basicamente utilizando a função *system()* e passando os parâmetros necessários. Entretanto, para algumas funções que necessitam funcionar paralelamente a outras será necessário criar processos filhos, como demonstrado no fluxograma da Fig. 9 que é o do relacionamento dos códigos desde o alarme sonoro e indicação visual ao reconhecimento facial e abertura da comporta.

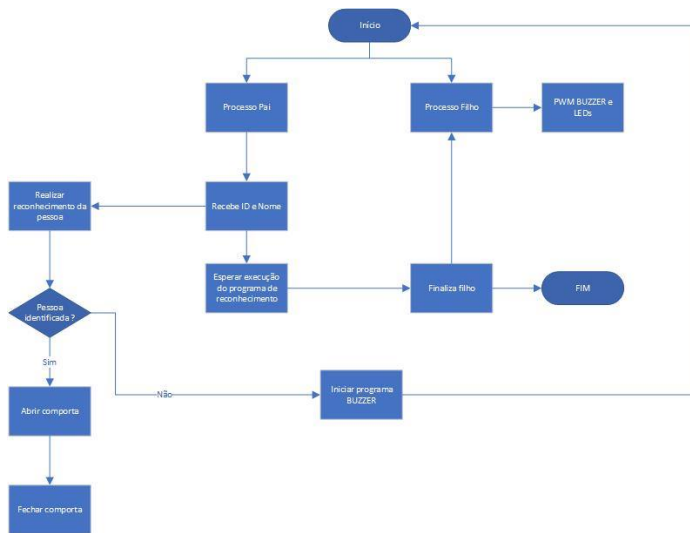


Fig. 9. Fluxograma do relacionamento dos códigos desde o alarme sonoro e indicação visual ao reconhecimento facial e abertura da comporta.

VII. RESULTADOS

Foi desenvolvido um dispositivo eletromecânico a partir da impressão na impressora 3D, o desenvolvimento do desenho foi a partir do *software Fusion* da *Autodesk*. Pode-se observar nas figuras seguintes como será o dispositivo para o gerenciamento de comprimidos:

Na Fig. 10 pode-se verificar como será a montagem, o mesmo terá diversos *slots* para inserir os comprimidos que suportará vários dias sem a necessidade de repor os comprimidos, os *slots* foi desenvolvido em forma circular para facilitar a rotação que será feita pelo motor de passo.

Também contará com uma base para coleta dos comprimidos que serão dispensados na hora que o usuário terá que ingeri-los, isso permitirá que o usuário sempre ingira os comprimidos adequados para seu tratamento na hora certa.

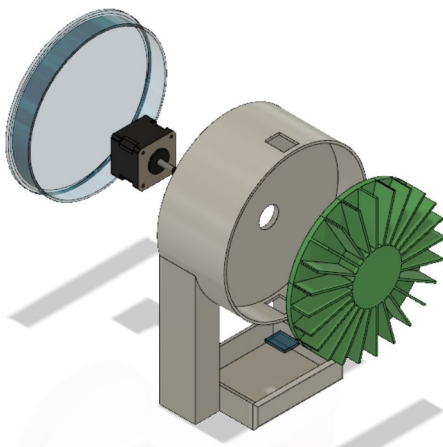


Fig. 10. Vista explodida do dispositivo

Na Fig. 11 e na Fig. 12 pode-se verificar o produto final:

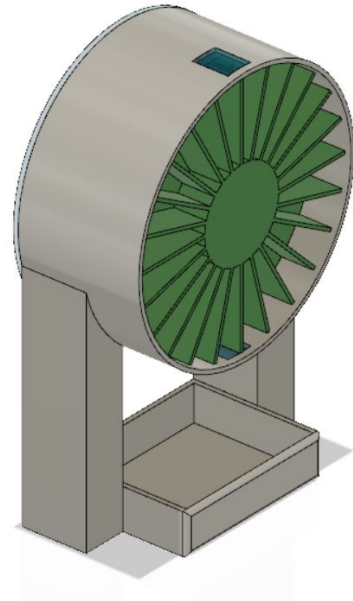


Fig. 11. Dispositivo eletromecânico para dispensar os comprimidos.

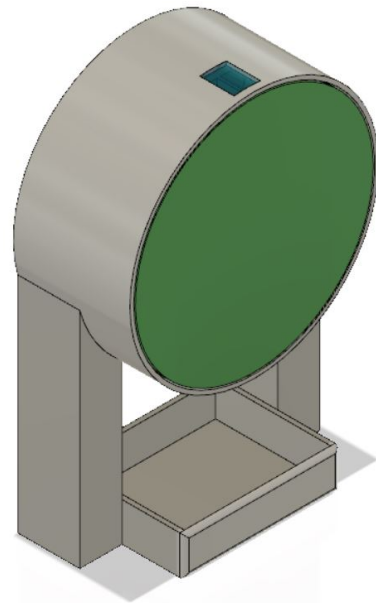


Fig. 12. Produto final.

A Fig. 13 mostra o resultado da implementação do do banco de dados, ao criar o programa pela primeira vez o sistema mostra que ainda não foi criado o arquivo, nas próximas vezes que o usuário fazer novos cadastramentos o arquivo mostrará que já existe um arquivo com todas as informações já cadastradas. O programa também comportará várias funcionalidades que poderá ser executado direto no terminal do programa, como ler o banco de dados, cadastrar novos usuários, mostrar dados do banco de dados, limpar o banco de dados, assim como a opção de fechar o programa.

```
**** arquivo nao existe ****

**** Qual modo de operacao deseja ****

**** Ler banco de dados : 1 ****

**** cadastrar dados : 2 ****

**** Mostrar dados : 3 ****

**** Limpar banco de dados : 4 ****

**** Fechar programa : 5 ****
```

Fig. 13. Opções para o usuário ter acesso ao sistema.

A Fig. 14 mostra o resultado da opção que o usuário tem de cadastrar usuários, no cadastramento o programa armazena no banco de dados algumas informações relevantes para o funcionamento do sistema como o nome do usuário como o seu nome, idade, quantidade de remédios a serem utilizados, nome dos remédios, quantidade de doses, como também os horários de cada remédio.

```
Modo selecionado : 2
Entre com a quantidade de pacientes: 1
Entre com o nome do usuario 1: Mateus
Usuario de ID numero : 1
Entre com a idade do usuario 1: 25
Entre com a quantidade de remedios do usuario 1: 1
Entre com o nome do remedio 1 do usuario 1: Dorflex
O slot do remedio 1 do usuario 1: 1
Entre com a quantidade de dose do remedio 1 do usuario 1: 1
Entre com a quantidade de horarios do remedio 1 do usuario 1: 1
Entre com a hora 1 do remedio 1 do usuario 1: 17
Entre com o minuto 1 do remedio 1 do usuario 1: 00
```

Fig. 14. Cadastramento do usuário.

A Fig. 14 mostra o resultado da opção que o usuário tem de visualizar todo o banco de dados direto no terminal.

```
usuario ID      idade  Quantidade de remedios  nome dos remedios  Slot dos remedios  Quantidade de dose  horario (h)
Mateus 1       25      1                      Dorflex 1         1                  17:00
```

Fig. 15. Banco de dados gerado (Arquivo).

A Fig. 16 mostra o resultado de quando o usuário se aproxima do dispositivo, sua imagem é capturada pela Webcam acoplada ao sistema e a partir deste ponto é realizado um processamento de imagem para identificar se a pessoa certa irá receber os comprimidos.

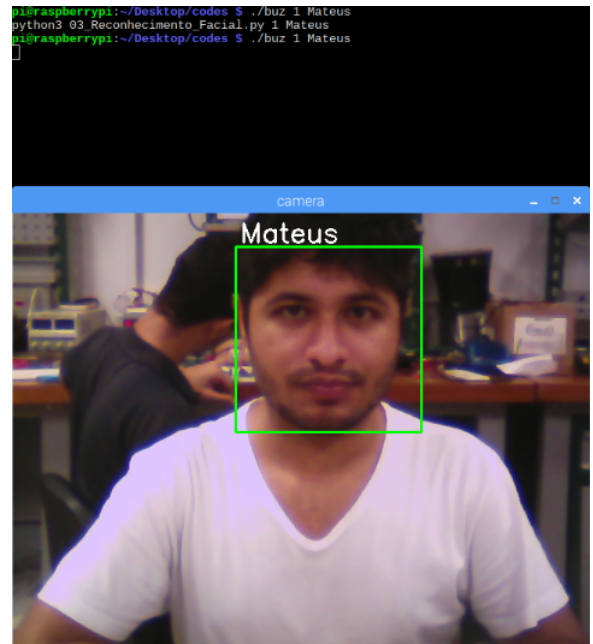


Fig. 16. Reconhecimento facial do usuário .

VIII. CONSIDERAÇÕES FINAIS

O projeto final apresentado alcançou os objetivos propostos. Foi implementado o banco de dados, o reconhecimento facial, a lógica dos motores, os leds e a implementação do relacionamento com o programa que reconhece a face com o programa de sinalização sonora do sistema.

Em vista do exposto, o projeto está de acordo com o que foi proposto para a matéria de sistemas embarcados, foi possível permitir ao administrador do sistema configurar a rotina de horários e quantidade de remédios a serem prescritos, além de apresentar um dispositivo eletromecânico para dispensar o remédio automaticamente por meio da *Raspberry pi*.

Os resultados parciais do desenvolvimento deste projeto promoveram uma confiança na finalização satisfatória deste dispositivo. Pôde-se desenvolver todos os códigos necessários para trabalhar isoladamente com cada componente da estrutura, restando desta forma apenas ao final integrá-los para o funcionamento sincronizado.

Foi possível aplicar os conhecimentos adquiridos durante a disciplina. Para o banco de dados, por exemplo, foi utilizado o conhecimento de arquivos visto no início do curso. Além disso, a comunicação entre os diversos componentes é feita pela troca de informações através de parâmetros no início da chamada do sistema onde é necessário a teoria da programação em C com ênfase em ponteiros. Adicionalmente, é possível observar a teoria de sistemas embarcados nas lógicas de programação empregadas em especial na criação de processos simultâneos.

O desenvolvimento do protótipo proporcionou a dupla ampliar o conhecimento da *Raspberry pi* e aplicar os vários conhecimentos adquiridos na disciplina durante todo o curso.

REFERÊNCIAS

- [1] O. P. Almeida, “Queixa de problemas com a memória e o diagnóstico de demência,” *Arq. Neuropsiquiatr.*, vol. 56, no. 3 A, pp. 412–418, 1998.
- [2] F. A. Cintra, M. E. Guariento, and L. A. Miyasaki, “Adesão medicamentosa em idosos em seguimento ambulatorial,” *Cien. Saude Colet.*, vol. 15, pp. 3507–3515, 2010.
- [3] Teixeira JJV, Spínola AWP. Comportamento do paciente idoso frente à aderência medicamentosa. *Arq Geriatr Gerontol* 1998; 2(1):5-9.
- [4] M. F. C. Kourouski and R. A. G. de Lima, “Adesão ao tratamento: vivências de adolescentes com hiv/aids,” *Rev Latino-am Enferm.*, vol. 17, no. 6, p. 111, 2009.
- [5] AMAZON, “MedFolio Wireless Pillbox (WP1050).” [Online]. Available: <https://www.amazon.com/Medfolio-MedFolio-Wireless-Pillbox-WP1050/dp/B00D3B7TVQ>.
- [6] The Sweethome, “The best smart pill dispenser (so far).” [Online]. Available: <https://www.engadget.com/2017/09/10/the-best-smart-pill-dispenser-so-far/>.
- [7] W. Siudzinski, “Spark Core powered automatic pill dispenser,” 2014. [Online]. Available: <https://suda.pl/spark-core-powered-automatic-pill-dispenser/>.
- [8] T. Nabelek and A. Nolte, “Automatic Pill Dispenser,” 2016. [Online]. Available: <https://www.youtube.com/watch?v=0gYyqYY8B-M>.

APÊNDICE - Implementação do Banco de Dados

```

1
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <string.h>
5
6
7 char usuario[10][40];
8 int idade[10];
9 int quantidade_remedio[10][1];
10 char remedio[10][10][20];
11 int quantidade_dose[10][10][1];
12 int quantidade_horario[10][10][1];
13 int hora[10][10][3];
14 int minuto[10][10][3];
15 int slot[10][40][1];
16 int id[10][1];
17 int id_slot=0;
18
19
20
21 char hora_lida1[1];
22 char minuto_lida1[1];
23 char hora_lida2[1];
24 char minuto_lida2[1];
25 char hora_lida4[1];
26 char minuto_lida4[1];
27 char idade_lida[3];
28 char remedio_lida[3];
29 char dose_lida[3];
30 char slot_lida[10];
31 char id_lida[10];
32
33
34 char leitura[10][400];
35 char *result;
36
37 int qtdp=0;
38 int r_qtdp=-1;
39 int i=0;
40 int j=0;
41 int a=0;
42 int selecao=0;
43 int inicializacao=0;

```

```

44 int escrever_dados();
45 int ler_dados();
46 int mostrar_dados();
47 int limpar_dados();
48 int chamar_codigo(int selecao);
49
50
51 // Criar ponteiro de arquivo
52 FILE *pont_arq;
53
54 int main(){
55
56
57
58 // abre arquivo
59 pont_arq=fopen("teste10.txt", "r");
60
61 if (pont_arq == NULL ){
62     printf("\n**** arquivo nao existe ****\n");
63     pont_arq=fopen("teste10.txt", "w");
64
65 // cabeCalho
66 fprintf(pont_arq, "%s", "usuario");
67 fprintf(pont_arq, "\t");
68 fprintf(pont_arq, "%s", "ID");
69 fprintf(pont_arq, "\t");
70 fprintf(pont_arq, "%s", "idade");
71 fprintf(pont_arq, "\t");
72 fprintf(pont_arq, "%s", "Quantidade de remedios");
73
74 fprintf(pont_arq, "\t");
75 fprintf(pont_arq, "%s", "nome dos remedios");
76 fprintf(pont_arq, "\t");
77 fprintf(pont_arq, "%s", "Slot dos remedios");
78 fprintf(pont_arq, "\t");
79 fprintf(pont_arq, "%s", "Quantidade de dose");
80 fprintf(pont_arq, "\t");
81 fprintf(pont_arq, "%s", "horario 01");
82 fprintf(pont_arq, "\t");
83 fprintf(pont_arq, "%s", "horario 02");
84 fprintf(pont_arq, "\t");
85 fprintf(pont_arq, "%s", "horario 03");
86 fprintf(pont_arq, "\n");
87
88 }else{
89     printf("\n**** arquivo aberto com sucesso ****\n");
90     ler_dados();
91     inicializacao=1;
92 };
93
94 fclose(pont_arq);
95
96 while (selecao!=5){
97     while (selecao<1 || selecao>5){
98         printf("\n**** Qual modo de operacao deseja ****\n");
99         printf("\n**** Ler banco de dados : 1 ****\n");
100         printf("\n**** cadastrar dados : 2 ****\n");
101         printf("\n**** Mostrar dados : 3 ****\n");
102         printf("\n**** Limpar banco de dados : 4 ****\n");
103         printf("\n**** Fechar programa : 5 ****\n");
104         scanf("%d", &selecao);
105     };
106
107     printf("\nModo selecionado : %d\n", selecao);
108
109     chamar_codigo(selecao);
110
111     fclose(pont_arq);
112
113     if (selecao<5 && selecao!=0){

```

```

114     selecao=0;
115 };
116
117 };
118 };
119
120
121
122
123 fclose(pont_arq);
124 printf("\nARQUIVO FECHADO\n");
125 getchar();
126
127 return (0);
128 }
129
130 int escrever_dados(){
131
132 int qtd=0;
133 int mem;
134
135 pont_arq=fopen("teste10.txt","w");
136
137 //*****
138 //prepara cabecalho do banco de dados
139 fprintf(pont_arq, "%s", "usuario");
140 fprintf(pont_arq, "\t");
141 fprintf(pont_arq, "%s", "ID");
142 fprintf(pont_arq, "\t");
143 fprintf(pont_arq, "%s", "idade");
144 fprintf(pont_arq, "\t");
145 fprintf(pont_arq, "%s", "Quantidade de remedios");
146
147 fprintf(pont_arq, "\t");
148 fprintf(pont_arq, "%s", "nome dos remedios");
149 fprintf(pont_arq, "\t");
150 fprintf(pont_arq, "%s", "Slot dos remedios");

```

```

150 fprintf(pont_arq, "\t");
151 fprintf(pont_arq, "%s", "Quantidade de dose");
152 fprintf(pont_arq, "\t");
153 fprintf(pont_arq, "%s", "horario 01");
154 fprintf(pont_arq, "\t");
155 fprintf(pont_arq, "%s", "horario 02");
156 fprintf(pont_arq, "\t");
157 fprintf(pont_arq, "%s", "horario 03");
158 fprintf(pont_arq, "\n");
159
160 //*****
161
162
163
164 //*****
165 //Inicio do cadastro de pacientes
166
167 printf("\nEntre com a quantidade de pacientes: ")
168 ;
169 scanf("%d", &qtd);
170 getchar();
171
172 mem=qtdp;
173 qtdp=qtd+qtdp;
174
175
176 for (i=0;i<qtdp;i++){ //i = numero do paciente
177
178 //
179 *****
180
181 //ZERAR VARIAVEIS TIPO CONTADORES DAS LOGICAS
182 DE CADA USUARIO
183 j=0;
184 a=0;
185 //

```



```

183 *****
184 //Nome do usuario
185 if(i>=mem || inicializacao<1){ //garante a nao
186     escrita sobre o dado anterior
187     printf("\nEntre com o nome do usuario %d: ",
188         (i+1));
189     gets(usuario[i]);
190 };
191 fprintf(pont_arq, "%s", usuario[i]);
192 fprintf(pont_arq, "\t");
193
194 //ID do usuario
195 if(i>=mem || inicializacao<1){ //garante a nao
196     escrita sobre o dado anterior
197     printf("\nUsuario de ID numero : %d", (i+1));
198     id[i][0]=i+1;
199     getchar();
200 };
201 fprintf(pont_arq, "%d", id[i][0]);
202 fprintf(pont_arq, "\t");
203
204 //Idade do usuario
205 if(i>=mem || inicializacao<1){ //garante a nao
206     escrita sobre o dado anterior
207     printf("\nEntre com a idade do usuario %d: ",
208         (i+1));
209     scanf("%d", &idade[i]);getchar();
210 };
211 fprintf(pont_arq, "%d", idade[i]);
212 fprintf(pont_arq, "\t");
213
214 //Quantidade de remedio
215 if(i>=mem || inicializacao<1){ //garante a nao
216     escrita sobre o dado anterior
217     printf("\nEntre com a quantidade de remedios
218         do usuario %d: ", (i+1));
219     scanf("%d", &quantidade_remedio[i][0]);
220     getchar();
221 };
222 fprintf(pont_arq, "%d", quantidade_remedio[i][0]);
223 fprintf(pont_arq, "\t");
224
225 for(j=0;j<quantidade_remedio[i][0];j++){ //laCo
226     do remedio
227
228     //Nome do remedio
229     if(i>=mem || inicializacao<1){ //garante a
230         nao escrita sobre o dado anterior
231         printf("\nEntre com o nome do remedio %d do
232             usuario %d: ", (j+1), (i+1));
233         gets(remedio[i][j]);
234     };
235     fprintf(pont_arq, "%s", remedio[i][j]);
236     fprintf(pont_arq, "\t");
237
238     //Slot do remedio
239     if(i>=mem || inicializacao<1){ //garante a
240         nao escrita sobre o dado anterior
241         id_slot++;
242         slot[i][j][0]=id_slot;
243         printf("\nO slot do remedio %d do usuario %
244             d: %d", (j+1), (i+1), slot[i][j][0]);
245         getchar();
246     };
247     fprintf(pont_arq, "%d", slot[i][j][0]);
248     fprintf(pont_arq, "\t");
249
250     //Quantidade de dose do remedio
251     if(i>=mem || inicializacao<1){ //garante a

```

```

241     nao escrita sobre o dado anterior
242     printf("\nEntre com a quantidade de dose do
243     remedio %d do usuario %d: ", (j+1), (i+1));
244     scanf("%d", &quantidade_dose[i][j][0]);
245     getchar();
246     };
247     fprintf(pont_arq, "%d", quantidade_dose[i][j]
248     ][0]);
249     fprintf(pont_arq, "\t");
250
251     //Quantidade de horarios do remedio
252     if(i>=mem || inicializacao<1){ //garante a
253     nao escrita sobre o dado anterior
254     printf("\nEntre com a quantidade de
255     horarios do remedio %d do usuario %d: ", (j+1),
256     (i+1));
257     scanf("%d", &quantidade_horario[i][j][0]);
258     getchar();
259     };
260
261     for(a=0;a<quantidade_horario[i][j][0];a++){
262     //Hora do remedio
263     if(i>=mem || inicializacao<1){ //garante a
264     nao escrita sobre o dado anterior
265     printf("\nEntre com a hora %d do remedio
266     %d do usuario %d: ", (a+1), (j+1), (i+1));
267     scanf("%d", &hora[i][j][a]);getchar();
268     };
269
270     //Minuto do remedio
271     if(i>=mem || inicializacao<1){ //garante a
272     nao escrita sobre o dado anterior
273     printf("\nEntre com o minuto %d do
274     remedio %d do usuario %d: ", (a+1), (j+1), (i
275     +1));
276     scanf("%d", &minuto[i][j][a]);getchar();
277     };
278
279     fprintf(pont_arq, "%d:%d", hora[i][j][a],
280     minuto[i][j][a]);
281     fprintf(pont_arq, "\t");
282     };
283
284     if(j!=(quantidade_remedio[i][0]-1)){
285     fprintf(pont_arq, "\n");
286     fprintf(pont_arq, "\t\t\t\t\t");
287     };
288     fprintf(pont_arq, "\n");
289     };
290
291     //Fim do cadastro de pacientes
292     //*****
293
294     inicializacao=1;
295     fclose(pont_arq);
296
297     return (0);
298 };
299
300 int ler_dados(){
301
302     //garantir variaveis zeradas
303     qtdp=0;
304     r_qtdp=-1;
305     i=0;
306     j=0;
307     a=0;
308     id_slot=0;
309
310     int cont=0;

```

```

301 int copia=0;
302 int mem[8];
303 int horario=0;
304 int laco=0;
305
306 pont_arq=fopen("teste10.txt","r");
307
308 while (!feof(pont_arq))
309 {
310 // Le uma linha (inclusive com o '\n')
311 result = fgets(leitura[r_qtdp], 400, pont_arq);
312 // o 'fgets' le ate 400 caracteres ou ate o '\n'
313
314 if (strlen(leitura[r_qtdp])<10 && inicializacao
315 ==1){ //se o arquivo resetou o db ele resete
316 inicializacao tbm
317 inicializacao=0;
318 };
319
320 if (result){ // Se foi possivel ler
321
322 if (leitura[r_qtdp][0]!='\t' && laco>0){
323 qtdp++;
324 copia=0;
325 cont=0;
326 mem[copia]=cont;
327
328 };
329 if (leitura[r_qtdp][0]=='\t' && leitura[r_qtdp]
330 [1]!='\t'){
331 copia=3;
332 cont=3;
333 mem[copia]=cont;
334 quantidade_remedio[qtdp][0]=
335 quantidade_remedio[qtdp][0]+1;
336 strcpy(hora_lida1,"0");

```

```

332 strcpy(minuto_lida1,"0");
333 strcpy(hora_lida2,"0");
334 strcpy(minuto_lida2,"0");
335 strcpy(hora_lida4,"0");
336 strcpy(minuto_lida4,"0");
337
338 };
339
340 while (leitura[r_qtdp][cont]!='\n' && r_qtdp!=-1
341 && strlen(leitura[r_qtdp])>10){
342
343 if (leitura[r_qtdp][cont]!='\t'){
344 switch (copia){
345 case 0:
346 usuario[qtdp-1][cont]=leitura[r_qtdp][
347 cont];
348
349 break;
350
351 case 1:
352 id_lida[cont-mem[copia]]=leitura[r_qtdp]
353 [cont]; //cont - mem elimina a posicao do \t e
354 copia os caracteres
355
356 break;
357
358 case 2:
359 if (atoi(id_lida)!=0){
360 id[qtdp-1][0]=atoi(id_lida); //grava
361 a variavel
362 strcpy(id_lida,"0"); //limpa a
363 variavel
364 };

```

```

363         idade_lida[cont-mem[copia]]=leitura[
364         r_qtdp][cont]; //cont - mem elimina a posicao
365         do \t e copia os caracteres
366
367         break;
368
369         case 3:
370             if(atoi(idade_lida)!=0){
371                 idade[qtdp-1]=atoi(idade_lida); //
372                 grava a variavel
373                 strcpy(idade_lida,"0"); //limpa a
374                 variavel
375             };
376
377             remedio_lida[cont-mem[copia]]=leitura[
378             r_qtdp][cont];
379
380             break;
381
382             case 4:
383                 if(atoi(remedio_lida)!=0){
384                     quantidade_remedio[qtdp-1][0]=atoi(
385                     remedio_lida);
386                     strcpy(remedio_lida,"0");
387                 };
388
389                 remedio[qtdp-1][quantidade_remedio[qtdp
390                 ][0]][cont-mem[copia]]=leitura[r_qtdp][cont];
391
392                 break;
393
394                 case 5:
395                     slot_lida[cont-mem[copia]]=leitura[
396                     r_qtdp][cont];
397
398                     break;
399
400

```

```

392         case 6:
393             if(atoi(slot_lida)!=0){
394                 slot[qtdp-1][quantidade_remedio[qtdp
395                 ][0]][0]=atoi(slot_lida);
396                 strcpy(slot_lida,"0");
397                 id_slot++; //incrementa o id dos
398                 slots ocupados
399             };
400
401             dose_lida[cont-mem[copia]]=leitura[
402             r_qtdp][cont];
403
404             break;
405
406             case 7:
407                 if(atoi(dose_lida)!=0){
408                     quantidade_dose[qtdp-1][
409                     quantidade_remedio[qtdp][0]][0]=atoi(dose_lida)
410                     ;
411                     strcpy(dose_lida,"0");
412                 };
413
414                 if(leitura[r_qtdp][cont]!=':'){
415                     if(horario==0){
416                         hora_lida1[cont-mem[copia]]=leitura
417                         [r_qtdp][cont];
418                     }else{
419                         minuto_lida1[cont-mem[copia]-3]=
420                         leitura[r_qtdp][cont];
421                     }
422                 }else{
423                     horario=1;
424                 };
425                 quantidade_horario[qtdp-1][

```

```

422     quantidade_remedio[qtdp][0][0]=1;
423     break;
424
425     case 8:
426         if( leitura[r_qtdp][cont]!=':'){
427             if(horario==0){
428                 hora_lida2[cont-mem[copia]]=leitura
429 [r_qtdp][cont];
430
431                 }else{
432                     minuto_lida2[cont-mem[copia]-3]=
433 leitura[r_qtdp][cont];
434
435                 }
436             }else{
437                 horario=1;
438             };
439             quantidade_horario[qtdp-1][
440 quantidade_remedio[qtdp][0][0]=2;
441 break;
442
443     case 9:
444         if( leitura[r_qtdp][cont]!=':'){
445             if(horario==0){
446                 hora_lida4[cont-mem[copia]]=leitura
447 [r_qtdp][cont];
448
449                 }else{
450                     minuto_lida4[cont-mem[copia]-3]=
451 leitura[r_qtdp][cont];
452
453                 }
454             }else{
455                 horario=1;
456             };
457             quantidade_horario[qtdp-1][
458 quantidade_remedio[qtdp][0][0]=3;

```

```

452         break;
453
454
455     }
456     }else{
457         copia++;
458         mem[copia]=cont+1;
459         horario=0;
460     }
461
462     cont++;
463
464 }
465 }
466
467
468 // *****
469 // conversao das horas para int
470 hora[qtdp-1][quantidade_remedio[qtdp][0][0]=
471 atoi(hora_lida1);
472 minuto[qtdp-1][quantidade_remedio[qtdp
473 ][0][0]=atoi(minuto_lida1);
474 hora[qtdp-1][quantidade_remedio[qtdp][0][1]=
475 atoi(hora_lida2);
476 minuto[qtdp-1][quantidade_remedio[qtdp
477 ][0][1]=atoi(minuto_lida2);
478 hora[qtdp-1][quantidade_remedio[qtdp][0][2]=
479 atoi(hora_lida4);
480 minuto[qtdp-1][quantidade_remedio[qtdp
481 ][0][2]=atoi(minuto_lida4);
482 // *****

```

```

483     laco++;
484     r_qtdp++;
485 }

```



```

483     inicializacao=1;
484     fclose(pont_arq);
485     return (0);
486
487 };
488
489 int mostrar_dados(){
490
491     printf("\n\nLeitura dos dados\n\n");
492     getchar();
493     int i=0;
494
495
496     //*****
497     //Inicio do leitura do cadastro de pacientes
498
499     for (i=0;i<qtdp;i++){ //i = numero do paciente
500
501         //
502         *****
503
504         //ZERAR VARIAVEIS TIPO CONTADORES DAS LOGICAS
505         //DE CADA USUARIO
506         j=0;
507         a=0;
508         //
509         *****
510
511         //Nome do usuario
512         printf("\nO nome do usuario %d e: %s", (i+1),
513         usuario[i]);
514         getchar();
515
516         //ID do usuario
517         printf("\nO usuario %d e: %d", (i+1), id[i][0])

```

```

514 ;
515 getchar();
516
517 //Idade do usuario
518 printf("\nA idade do usuario %d e: %d anos", (i
519 +1), idade[i]);
520 getchar();
521
522 //Quantidade de remedio
523 printf("\nA quantidade de remedios do usuario %
524 d e: %d ", (i+1), quantidade_remedio[i][0]);
525 getchar();
526
527 for(j=0;j<quantidade_remedio[i][0];j++){ //laco
528 do remedio
529
530     //Nome do remedio
531     printf("\nO nome do remedio %d do usuario %d
532 e: %s", (j+1), (i+1), remedio[i][j]);
533     getchar();
534
535     //Slot do remedio
536     printf("\nO slot do remedio %d do usuario %d
537 e: %d", (j+1), (i+1), slot[i][j][0]);
538     getchar();
539
540     //Quantidade de dose do remedio
541     printf("\nA quantidade de dose do remedio %d
542 do usuario %d e: %d", (j+1), (i+1),
543     quantidade_dose[i][j][0]);
544     getchar();
545
546     //Quantidade de horarios do remedio
547     printf("\nA quantidade de horarios do remedio
548 %d do usuario %d e: %d", (j+1), (i+1),
549     quantidade_horario[i][j][0]);
550     getchar();

```

```

541         for(a=0;a<quantidade_horario[i][j][0];a++){
542             //Hora do remedio
543             printf("\nO horario %d do remedio %d do
544             usuario %d e: %d : %d", (a+1), (j+1), (i+1),
545             hora[i][j][a], minuto[i][j][a]);
546             getchar();
547         };
548
549
550     };
551
552 };
553
554
555 };
556
557 //Fim da leitura do cadastro de pacientes
558 //*****
559
560 return (0);
561 };
562
563 int limpar_dados(){
564 pont_arq=fopen("teste10.txt","w");
565
566 //*****
567 //prepara cabecalho do banco de dados
568 fprintf(pont_arq, "%s", "usuario");
569 fprintf(pont_arq, "\t");
570 fprintf(pont_arq, "%s", "idade");
571 fprintf(pont_arq, "\t");
572 fprintf(pont_arq, "%s", "Quantidade de remedios
573 ");
574 fprintf(pont_arq, "\t");
575
576 fprintf(pont_arq, "%s", "nome dos remedios");
577 fprintf(pont_arq, "\t");
578 fprintf(pont_arq, "%s", "Quantidade de dose");
579 fprintf(pont_arq, "\t");
580 fprintf(pont_arq, "%s", "horario 01");
581 fprintf(pont_arq, "\t");
582 fprintf(pont_arq, "%s", "horario 02");
583 fprintf(pont_arq, "\t");
584 fprintf(pont_arq, "%s", "horario 03");
585 fprintf(pont_arq, "\n");
586 //*****
587
588 fclose(pont_arq);
589 return (0);
590 };
591
592 int chamar_codigo(int selecao){
593
594
595 switch (selecao){
596     case 1:
597         ler_dados();
598         break;
599     case 2:
600         escrever_dados();
601         break;
602     case 3:
603         mostrar_dados();
604         break;
605     case 4:
606         limpar_dados();
607         break;
608 };
609
610 return (0);
611 };

```

APÊNDICE - Implementação do código para acionamento do motor de passo

```
1 #include <stdio.h>
2 #include <wiringPi.h>
3 #include <stdlib.h>
4
5 int in1 = 7;
6 int in2 = 15;
7 int in3 = 0;
8 int in4 = 1;
9
10 const int calPasso = 18;
11 const int del = 5;
12
13 int passo = 0;
14
15 int movimenta(int passos, int direcao){
16
17     if (direcao == 1 )
18     {
19         for (int i=0; i<passos; i++)
20         {
21             switch(passo)
22             {
23                 case 4:
24                 case 0:
25                     digitalWrite(in1, HIGH);
26                     digitalWrite(in2, LOW);
27                     digitalWrite(in3, LOW);
28                     digitalWrite(in4, HIGH);
29                     delay(del);
30                     passo = 1;
31                 break;
32                 case 1:
33                     digitalWrite(in1, LOW);
34                     digitalWrite(in2, HIGH);
```

```
35                     digitalWrite(in3, LOW);
36                     digitalWrite(in4, HIGH);
37                     delay(del);
38                     passo = 2;
39                 break;
40                 case 2:
41                     digitalWrite(in1, LOW);
42                     digitalWrite(in2, HIGH);
43                     digitalWrite(in3, HIGH);
44                     digitalWrite(in4, LOW);
45                     delay(del);
46                     passo = 3;
47                 break;
48                 case 3:
49                     digitalWrite(in1, HIGH);
50                     digitalWrite(in2, LOW);
51                     digitalWrite(in3, HIGH);
52                     digitalWrite(in4, LOW);
53                     delay(del);
54                     passo = 4;
55                 break;
56             }
57         }
58     }
59     else if (direcao == 2)
60     {
61         for (int i=0; i<passos; i++)
62         {
63             switch(passo)
64             {
65                 case 0:
66                 case 1:
67                     digitalWrite(in1, HIGH);
68                     digitalWrite(in2, LOW);
69                     digitalWrite(in3, HIGH);
70                     digitalWrite(in4, LOW);
71                     delay(del);
```

```

72         passo = 4;
73         break;
74     case 2:
75         digitalWrite(in1, HIGH);
76         digitalWrite(in2, LOW);
77         digitalWrite(in3, LOW);
78         digitalWrite(in4, HIGH);
79         delay(del);
80         passo = 1;
81         break;
82     case 3:
83         digitalWrite(in1, LOW);
84         digitalWrite(in2, HIGH);
85         digitalWrite(in3, LOW);
86         digitalWrite(in4, HIGH);
87         delay(del);
88         passo = 2;
89         break;
90     case 4:
91         digitalWrite(in1, LOW);
92         digitalWrite(in2, HIGH);
93         digitalWrite(in3, HIGH);
94         digitalWrite(in4, LOW);
95         delay(del);
96         passo = 3;
97         break;
98     }
99 }
100 }
101 }
102     return passo;
103 }
104
105 int main(int argc, char **argv)
106 {
107
108

```

```

109 if(wiringPiSetup()==-1)
110 { puts("Deu ruim no setup");
111   return -1;
112 }
113
114 pinMode(in1, OUTPUT);
115 pinMode(in2, OUTPUT);
116 pinMode(in3, OUTPUT);
117 pinMode(in4, OUTPUT);
118
119 int counter = atoi(argv[1]);
120 while(counter!=0)
121 {
122     passo = movimenta(calPasso,1);
123     counter--;
124 }
125 }

```

APÊNDICE - Implementação do código para acionamento do servo motor

```

1 #include <wiringPi.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 int in5 = 5;
6
7 void servo0graus()
8 {
9     digitalWrite(in5, HIGH);
10    delayMicroseconds(600);
11    digitalWrite(in5, LOW);
12    for (int i=0; i<32;i++) delayMicroseconds(600);
13 }
14
15 void servoPIgraus()
16 {

```

```

17 digitalWrite(in5, HIGH);
18 delayMicroseconds(600);
19 digitalWrite(in5, LOW);
20 for (int i=0; i<26;i++) delayMicroseconds(600);
21 }
22
23
24 void servo90graus()
25 {
26     digitalWrite(in5, HIGH);
27     delayMicroseconds(1500);
28     digitalWrite(in5, LOW);
29     for (int i=0; i<12;i++) delayMicroseconds(1500);
30 }
31 void servo180graus()
32 {
33     digitalWrite(in5, HIGH);
34     delayMicroseconds(2400);
35     digitalWrite(in5, LOW);
36     for (int i=0; i<7;i++) delayMicroseconds(2400);
37 }
38
39 int main()
40 {
41     if(wiringPiSetup()==-1)
42     {
43         puts("Deu ruim no setup");
44         return -1;
45     }
46
47     pinMode(in5, OUTPUT);
48
49     for( int i=0; i<100;i++) servo180graus();
50     delay(500);
51     for(int i=0; i<100;i++) servoPIgraus();
52     return 0;
53 }

```

APÊNDICE - Implementação do código para o aviso ao usuário (alarme)

```

1
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <string.h>
5 #include <time.h>
6
7 // *****
8 //CASO ALTERE A QUANTIDADE DE SLOTS ... ALTERAR
9 //AQUI
10
11 int qtd_slots=24;
12
13 // *****
14
15 char usuario[10][40];
16 int idade[10];
17 int quantidade_remedio[10][1];
18 char remedio[10][10][20];
19 int quantidade_dose[10][10][1];
20 int quantidade_horario[10][10][1];
21 int hora[10][10][3];
22 int minuto[10][10][3];
23 int slot[10][40][1];
24 int id[10][1];
25 int id_slot=0;
26 int checar_horario;
27
28
29
30 char hora_lida1[1];
31 char minuto_lida1[1];
32 char hora_lida2[1];
33 char minuto_lida2[1];

```



```

34 char hora_lida4[1];
35 char minuto_lida4[1];
36 char idade_lida[3];
37 char remedio_lida[3];
38 char dose_lida[3];
39 char slot_lida[10];
40 char id_lida[10];
41
42 //*****
43 //VARIABLES PARA LER AS HORAS DO SISTEMA
44 char horario_sistema[9];
45 char hora_sistema[1];
46 char minuto_sistema[1];
47 char minuto_anterior[1]; //fara a verificacao a
    cada minuto
48 int hora_check=0;
49 int minuto_check=0;
50
51 //*****
52
53 char leitura[10][400];
54 char *result;
55
56 int qtdp=0;
57 int r_qtdp=-1;
58 int i=0;
59 int j=0;
60 int a=0;
61 int selecao=0;
62 int inicializacao=0;
63
64
65 int sistema();
66 int checar_dados();
67
68 //Criar ponteiro de arquivo
69 FILE *pont_arq;

```

```

70
71 int main(){
72
73
74
75 //abre arquivo
76 pont_arq=fopen("teste10.txt","r");
77
78 if (pont_arq == NULL ){
79     printf("\n**** arquivo nao existe ****\n");
80     pont_arq=fopen("teste10.txt","w");
81
82 //cabecalho
83 fprintf(pont_arq, "%s", "usuario");
84 fprintf(pont_arq, "\t");
85 fprintf(pont_arq, "%s", "ID");
86 fprintf(pont_arq, "\t");
87 fprintf(pont_arq, "%s", "idade");
88 fprintf(pont_arq, "\t");
89 fprintf(pont_arq, "%s", "Quantidade de remedios
    ");
90 fprintf(pont_arq, "\t");
91 fprintf(pont_arq, "%s", "nome dos remedios");
92 fprintf(pont_arq, "\t");
93 fprintf(pont_arq, "%s", "Slot dos remedios");
94 fprintf(pont_arq, "\t");
95 fprintf(pont_arq, "%s", "Quantidade de dose");
96 fprintf(pont_arq, "\t");
97 fprintf(pont_arq, "%s", "horario 01");
98 fprintf(pont_arq, "\t");
99 fprintf(pont_arq, "%s", "horario 02");
100 fprintf(pont_arq, "\t");
101 fprintf(pont_arq, "%s", "horario 03");
102 fprintf(pont_arq, "\n");
103
104 }else{
105     printf("\n**** arquivo aberto com sucesso

```

```

106     ****\n");
107     inicializacao=1;
108 };
109 fclose(pont_arq);
110
111 while (selecao!=5){
112     sistema();
113
114     if (checar_horario==1){
115         strcpy(minuto_anterior,minuto_sistema);
116         checar_dados();
117     };
118 };
119
120
121
122
123
124
125
126 fclose(pont_arq);
127 printf("\nARQUIVO FECHADO\n");
128 getchar();
129
130 return (0);
131 }
132
133
134
135 int checar_dados(){
136
137 //garantir variaveis zeradas
138 qtdp=0;
139 r_qtdp=-1;
140 i=0;
141 j=0;

```

```

142 a=0;
143 id_slot=0;
144
145
146
147 int cont=0;
148 int copia=0;
149 int mem[8];
150 int horario=0;
151 int laco=0;
152
153
154
155 pont_arq=fopen("teste10.txt","r");
156
157 while (!feof(pont_arq))
158 {
159     // Le uma linha (inclusive com o '\n')
160     result = fgets(leitura[r_qtdp], 400, pont_arq);
161     // o 'fgets' le ate 400 caracteres ou ate o '\n'
162
163     if (strlen(leitura[r_qtdp])<10 && inicializacao
164 ==1){ //se o arquivo resetou o db ele resete
165         inicializacao tbm
166         inicializacao=0;
167     };
168
169     if (result){ // Se foi possivel ler
170
171         if (leitura[r_qtdp][0]!='\t' && laco>0){
172             qtdp++;
173             //printf("\nA quantidade de paciente eh : %d\n", qtdp);getchar();
174             copia=0;
175             cont=0;
176             mem[copia]=cont;

```

```

174     };
175
176     if (leitura[r_qtdp][0] == '\t' && leitura[r_qtdp][1] == '\t') {
177         copia = 3;
178         cont = 3;
179         mem[copia] = cont;
180         quantidade_remedio[qtdp][0] =
181             quantidade_remedio[qtdp][0] + 1;
182         strcpy(hora_lida1, "0");
183         strcpy(minuto_lida1, "0");
184         strcpy(hora_lida2, "0");
185         strcpy(minuto_lida2, "0");
186         strcpy(hora_lida4, "0");
187         strcpy(minuto_lida4, "0");
188     };
189
190
191     // printf("\nPaciente %d : %s", (qtdp), leitura[r_qtdp]);
192
193
194     while (leitura[r_qtdp][cont] != '\n' && r_qtdp != -1
195           && strlen(leitura[r_qtdp]) > 10) {
196
197         if (leitura[r_qtdp][cont] != '\t') {
198             switch (copia) {
199                 case 0:
200                     usuario[qtdp - 1][cont] = leitura[r_qtdp][cont];
201
202                     // printf("\nO caractere usuario copiado eh : %c\n", usuario[qtdp - 1][cont]);
203                     // printf("\nO cursor eh : %d\n", cont);
204                     getchar();
205                     break;

```

```

204         case 1:
205             id_lida[cont - mem[copia]] = leitura[r_qtdp][cont]; // cont - mem elimina a posicao do \t e copia os caracteres
206             // printf("\nO caractere idade copiado eh : %c\n", idade_lida[cont - mem[copia]]);
207             getchar();
208             // printf("\nO cursor eh : %d\n", cont);
209             getchar();
210             break;
211
212         case 2:
213             if (atoi(id_lida) != 0) {
214                 id[qtdp - 1][0] = atoi(id_lida); // grava a variavel
215                 strcpy(id_lida, "0"); // limpa a variavel
216             };
217
218             idade_lida[cont - mem[copia]] = leitura[r_qtdp][cont]; // cont - mem elimina a posicao do \t e copia os caracteres
219             // printf("\nO caractere idade copiado eh : %c\n", idade_lida[cont - mem[copia]]);
220             getchar();
221             // printf("\nO cursor eh : %d\n", cont);
222             getchar();
223             break;
224
225         case 3:
226             if (atoi(idade_lida) != 0) {
227                 idade[qtdp - 1] = atoi(idade_lida); // grava a variavel
228                 strcpy(idade_lida, "0"); // limpa a variavel

```

```

227         };
228
229         remedio_lida[cont-mem[copia]]= leitura [
230         r_qtdp ][ cont ];
231         // printf("\nO caracter QTD REMEDIO
232         copiado eh : %c\n", remedio_lida[cont-mem[copia
233         ]]); getchar();
234         // printf("\nO cursor eh : %d\n", cont);
235         getchar();
236         break;
237
238         case 4:
239         if ( atoi( remedio_lida ) != 0 ) {
240             quantidade_remedio[ qtdp - 1 ][ 0 ] = atoi (
241             remedio_lida );
242             strcpy ( remedio_lida , "0" );
243         };
244
245         remedio[ qtdp - 1 ][ quantidade_remedio[ qtdp
246         ][ 0 ] ][ cont - mem[ copia ] ] = leitura [ r_qtdp ][ cont ];
247         // printf("\nO caracter NOME REMEDIO
248         copiado eh : %c\n", remedio[ qtdp - 1 ][
249         quantidade_remedio[ qtdp ][ 0 ] ][ cont - mem[ copia ] ]);
250         getchar();
251         // printf("\nO cursor eh : %d\n", cont);
252         getchar();
253         break;
254
255         case 5:
256         slot_lida[cont-mem[copia]]= leitura [
257         r_qtdp ][ cont ];
258         // printf("\nO caracter qtd dose copiado
259         eh : %c\n", dose_lida[cont-mem[copia]]);
260         getchar();
261         // printf("\nO cursor eh : %d\n", cont);
262         getchar();
263         break;
264
265         case 6:
266         slot_lida[cont-mem[copia]]= leitura [
267         r_qtdp ][ cont ];
268         // printf("\nO caracter qtd dose copiado
269         eh : %c\n", dose_lida[cont-mem[copia]]);
270         getchar();
271         // printf("\nO cursor eh : %d\n", cont);
272         getchar();
273         break;
274
275         case 7:
276         if ( atoi( dose_lida ) != 0 ) {
277             quantidade_dose[ qtdp - 1 ][
278             quantidade_remedio[ qtdp ][ 0 ] ][ 0 ] = atoi ( dose_lida )
279             ;
280             strcpy ( dose_lida , "0" );
281         };
282
283         if ( leitura [ r_qtdp ][ cont ] != ':' ) {
284             if ( horario == 0 ) {
285                 hora_lida1[cont-mem[copia]]= leitura
286                 [ r_qtdp ][ cont ];
287                 // printf("\nO caracter horal
288                 copiado eh : %c\n", hora_lida1[cont-mem[copia
289                 ]]); getchar();
290                 // printf("\nO cursor eh : %d\n",
291                 cont); getchar(); printf("\nO cursor eh : %d\n",

```

```

250         break;
251
252         case 6:
253         if ( atoi( slot_lida ) != 0 ) {
254             slot[ qtdp - 1 ][ quantidade_remedio[ qtdp
255             ][ 0 ] ][ 0 ] = atoi ( slot_lida );
256             strcpy ( slot_lida , "0" );
257             id_slot++; // incrementa o id dos
258             slots ocupados
259         };
260
261         dose_lida[cont-mem[copia]]= leitura [
262         r_qtdp ][ cont ];
263         // printf("\nO caracter qtd dose copiado
264         eh : %c\n", dose_lida[cont-mem[copia]]);
265         getchar();
266         // printf("\nO cursor eh : %d\n", cont);
267         getchar();
268         break;
269
270         case 7:
271         if ( atoi( dose_lida ) != 0 ) {
272             quantidade_dose[ qtdp - 1 ][
273             quantidade_remedio[ qtdp ][ 0 ] ][ 0 ] = atoi ( dose_lida )
274             ;
275             strcpy ( dose_lida , "0" );
276         };
277
278         if ( leitura [ r_qtdp ][ cont ] != ':' ) {
279             if ( horario == 0 ) {
280                 hora_lida1[cont-mem[copia]]= leitura
281                 [ r_qtdp ][ cont ];
282                 // printf("\nO caracter horal
283                 copiado eh : %c\n", hora_lida1[cont-mem[copia
284                 ]]); getchar();
285                 // printf("\nO cursor eh : %d\n",
286                 cont); getchar(); printf("\nO cursor eh : %d\n",

```

```

275     cont);getchar();
276         }else{
277             minuto_lida1[cont-mem[copia]-3]=
278             leitura[r_qtdp][cont];
279             // printf("\nO caracter minuto1
280             copiado eh : %c\n", minuto_lida1[cont-mem[copia]
281             -3]);getchar();
282             // printf("\nO cursor eh : %d\n",
283             cont);getchar();
284         }
285         }else{
286             horario=1;
287         };
288         quantidade_horario[qtdp-1][
289         quantidade_remedio[qtdp][0]][0]=1;
290         break;
291
292         case 8:
293             if(leitura[r_qtdp][cont]!=':'){
294                 if(horario==0){
295                     hora_lida2[cont-mem[copia]]=leitura
296                     [r_qtdp][cont];
297                     // printf("\nO caracter hora2
298                     copiado eh : %c\n", hora_lida2[cont-mem[copia]
299                     ]);getchar();
300                     // printf("\nO cursor eh : %d\n",
301                     cont);getchar();
302                 }else{
303                     minuto_lida2[cont-mem[copia]-3]=
304                     leitura[r_qtdp][cont];
305                     // printf("\nO caracter minuto2
306                     copiado eh : %c\n", minuto_lida2[cont-mem[copia]
307                     -3]);getchar();
308                     // printf("\nO cursor eh : %d\n",
309                     cont);getchar();
310                 }
311             }else{
312                 minuto_lida2[cont-mem[copia]-3]=
313                 leitura[r_qtdp][cont];
314                 // printf("\nO caracter minuto2
315                 copiado eh : %c\n", minuto_lida2[cont-mem[copia]
316                 -3]);getchar();
317                 // printf("\nO cursor eh : %d\n",
318                 cont);getchar();
319             }
320         }
321     }else{
322

```

```

298         horario=1;
299     };
300     quantidade_horario[qtdp-1][
301     quantidade_remedio[qtdp][0]][0]=2;
302     break;
303
304     case 9:
305         if(leitura[r_qtdp][cont]!=':'){
306             if(horario==0){
307                 hora_lida4[cont-mem[copia]]=leitura
308                 [r_qtdp][cont];
309                 // printf("\nO caracter hora3
310                 copiado eh : %c\n", hora_lida4[cont-mem[copia]
311                 ]);getchar();
312                 // printf("\nO cursor eh : %d\n",
313                 cont);getchar();
314             }else{
315                 minuto_lida4[cont-mem[copia]-3]=
316                 leitura[r_qtdp][cont];
317                 // printf("\nO caracter minuto3
318                 copiado eh : %c\n", minuto_lida4[cont-mem[copia]
319                 -3]);getchar();
320                 // printf("\nO cursor eh : %d\n",
321                 cont);getchar();
322             }
323         }else{
324             horario=1;
325         };
326         quantidade_horario[qtdp-1][
327         quantidade_remedio[qtdp][0]][0]=3;
328         break;
329
330     }
331 }else{
332     copia++;
333     mem[copia]=cont+1;

```



```

325         horario=0;
326     }
327
328     cont++;
329
330 }
331 }
332
333
334     // *****
335     // conversao das horas para int
336     hora[qtdp-1][quantidade_remedio[qtdp][0]][0]=
337     atoi(hora_lida1);
338     minuto[qtdp-1][quantidade_remedio[qtdp][0]][0]=
339     atoi(hora_lida1);
340     hora[qtdp-1][quantidade_remedio[qtdp][0]][1]=
341     atoi(hora_lida2);
342     minuto[qtdp-1][quantidade_remedio[qtdp][0]][1]=
343     atoi(minuto_lida2);
344     hora[qtdp-1][quantidade_remedio[qtdp][0]][2]=
345     atoi(hora_lida4);
346     minuto[qtdp-1][quantidade_remedio[qtdp][0]][2]=
347     atoi(minuto_lida4);
348     // *****
349
350     // *****
351     // Checar horarios dos remedios
352     if(hora_check==hora[qtdp-1][
353     quantidade_remedio[qtdp][0]][0] && minuto_check
354     ==minuto[qtdp-1][quantidade_remedio[qtdp][0]][0]){
355
356         printf("\nO usuario %s deve tomar o
357         remedio %s\n", usuario[qtdp-1], remedio[qtdp-1][
358         quantidade_remedio[qtdp][0]]);
359         //INSERIR AQUI O COMANDO PARA A RASPBERRY
360         =)))
361     };
362     if(hora_check==hora[qtdp-1][
363     quantidade_remedio[qtdp][0]][2] && minuto_check
364     ==minuto[qtdp-1][quantidade_remedio[qtdp][0]][2]){
365
366         printf("\nO usuario %s deve tomar o
367         remedio %s\n", usuario[qtdp-1], remedio[qtdp-1][
368         quantidade_remedio[qtdp][0]]);
369         //INSERIR AQUI O COMANDO PARA A RASPBERRY
370         =)))
371     };
372
373     // *****
374
375     laco++;
376     r_qtdp++;
377 }
378
379 inicializacao=1;

```

```
375 fclose(pont_arq);
376 return (0);
377
378 };
379
380
381
382
383 int sistema(){
384
385
386     _strtime(horario_sistema);
387
388     hora_sistema[0]=horario_sistema[0];
389     hora_sistema[1]=horario_sistema[1];
390     hora_sistema[2]='\0';
391     hora_check=atoi(hora_sistema);
392     minuto_sistema[0]=horario_sistema[3];
393     minuto_sistema[1]=horario_sistema[4];
394     minuto_sistema[2]='\0';
395     minuto_check=atoi(minuto_sistema);
396
397
398     if(atoi(minuto_anterior)!=atoi(minuto_sistema)){
399         checar_horario=1;
400     }else{
401         checar_horario=0;
402     };
403
404     return (0);
405 };
```