Pi-Mirinha

Assistente de Cozinha

Amauri da Costa Júnior
Engenharia Eletrônica
UNB-FGA
Brasília, Brasil
amauri_cj@hotmail.com

Thássio Gabriel Farias dos Santos
Engenharia Eletrônica
UNB-FGA
Brasília, Brasil
thassio.96@gmail.com

Resumo — Cozinhar está na moda e se tornou algo que muitas pessoas querem aprender, pensando nisso, nós propomos criar um assistente de cozinha. Este assistente consegue falar com você dizendo quais os ingredientes devem ser utilizados naquela receita e te dizer como executar os passos do modo de preparo. Para isso iremos utilizar uma raspberry Pi para que ela consiga executar o que foi proposto.

Palavras-chave — ; Cozinhar, aprender, assistente, raspberry pi

I. Introdução

Cozinhar está na moda, para confirmar isso é só ligar a televisão e ver que existe uma grande variedades de programas culinários que fazem muito sucesso em diversos formatos, procurando também na internet é possível encontrar muito conteúdo sobre isto, gerando um interesse crescente nas pessoas[1]. Pensando nisso, decidimos criar um assistente de cozinha que tem o intuito de auxiliar cozinheiros que estão começando agora a replicar receitas simples. Para que isso seja possível utilizaremos uma Raspberry pi modelo 3 B, ela em complemento a acessórios como displays, alto falantes e microfones, atende aos requisitos do nosso projeto[2].

Produtos semelhantes como panelas inteligentes fazem todo o trabalho para o usuário ao invés de auxiliar nas tarefa de cozinhar[3]. Existem no mercado também aplicativos para smartphones e tablets que apresentam as receitas para o usuário facilitando a tarefa de cozinhar[4].

A forma com a qual iremos fazer com que a raspberry consiga falar as instruções para o usuário é através de um sintetizador de voz. Este sintetizador é conhecido como Text to speech, ou seja, ele transforma texto em voz, assim, é possível que a receita seja lida[5]. O usuário poderá controlar o programa pela tela touch, ou então, para que não precise desocupar suas mãos do preparo da comida, a leitura do modo de preparo também poderá ser controlada com comandos de voz[6].

II. Desenvolvimento

A. Descrição de Hardware

Para a realização deste trabalho, foi utilizada a seguinte lista de materiais

Bill of materials			
Item	Unidade	Quantidade	Preço (Reais)
Raspberry pi 3B	cada	1	198,69
Alto falante com circuito amplificador	cada	1	20,00
Display	cada	1	115,00
Microfone USB	cada	1	60,35
Fonte 2,1 A	cada	1	25,00

O controlador é a raspberry pi 3B, que possui um módulo wifi integrado e uma interface gráfica, características extremamente necessárias para o desenvolvimento do projeto.

O display utilizado é uma tela LCD de 3.5 polegadas com tela touchscreen, a comunicação entre a raspberry e o display é feita através de 26 pinos GPIO, que apesar de possuir uma transmissão lenta de dados, não interfere no projeto pois não são necessárias respostas rápidas. A tela touchscreen será utilizada como uma das opções para navegar pelos menus do programa que organiza as receitas[7].

B. Descrição de Software

O código desenvolvido foi feito com base nas linguagens C e C++ com auxílio da framework QT para desenvolvimento da interface gráfica. Programas complementares que irão rodar com nosso software são instalados através dos pacotes gTTS, mpg123 e Snips. Com o gTTS é possível que ocorra a sintetização de voz, para que seja possível tocar o áudio gerado pelo sintetizador de voz, utilizamos o mpg123 [8][9]. Através do Snips é possível salvar comandos simples, para que quando esses forem falados pelo usuário, este possa decidir quando passar para o próximo passo da receita, ou então voltar em um passo anterior[10].

O programa possui uma interface gráfica que tem como objetivo permitir que o usuário controle todo o processo de forma intuitiva e prática. Há códigos feitos em C que são executados pelo programa em C++ através de chamadas de terminal, estas tarefas escritas em C executam tarefas específicas como adicionar receitas no banco de receitas e processar informações contidas nas receitas. O programa em C++ também administra a utilização dos programas de Text to speech e execução de áudio. O funcionamento do programa atual pode ser melhor compreendido através da leitura dos fluxogramas encontrados nos Anexos 1,2,3,4 e 5, o funcionamento final do software do projeto pode ser visto no anexo 6.

III. Resultados

A linguagem C++ em conjunto com a framework QT conseguiu suprir as necessidades para o desenvolvimento de uma interface gráfica, pois foi possível criar uma interface de fácil uso na qual o usuário consegue facilmente manipular o software de forma a conseguir executar tudo o que foi proposto no projeto.

O programa principal consegue facilmente pedir para que programas em C e os instalados por pacotes executem as tarefas as quais os foram designadas como adicionar receitas ao banco de receitas, processar os dados da receita, sintetizar voz e tocar o áudio da voz sintetizada.

Os programas em C e os instalados funcionam de acordo com o esperado.

A parte do reconhecimento de voz apesar de não estar implementada ao nosso software, já conseguimos fazer com que ele reconheça comandos que iremos utilizar para pedir que o programa repita, passe ou volte a instrução.

IV. Conclusão

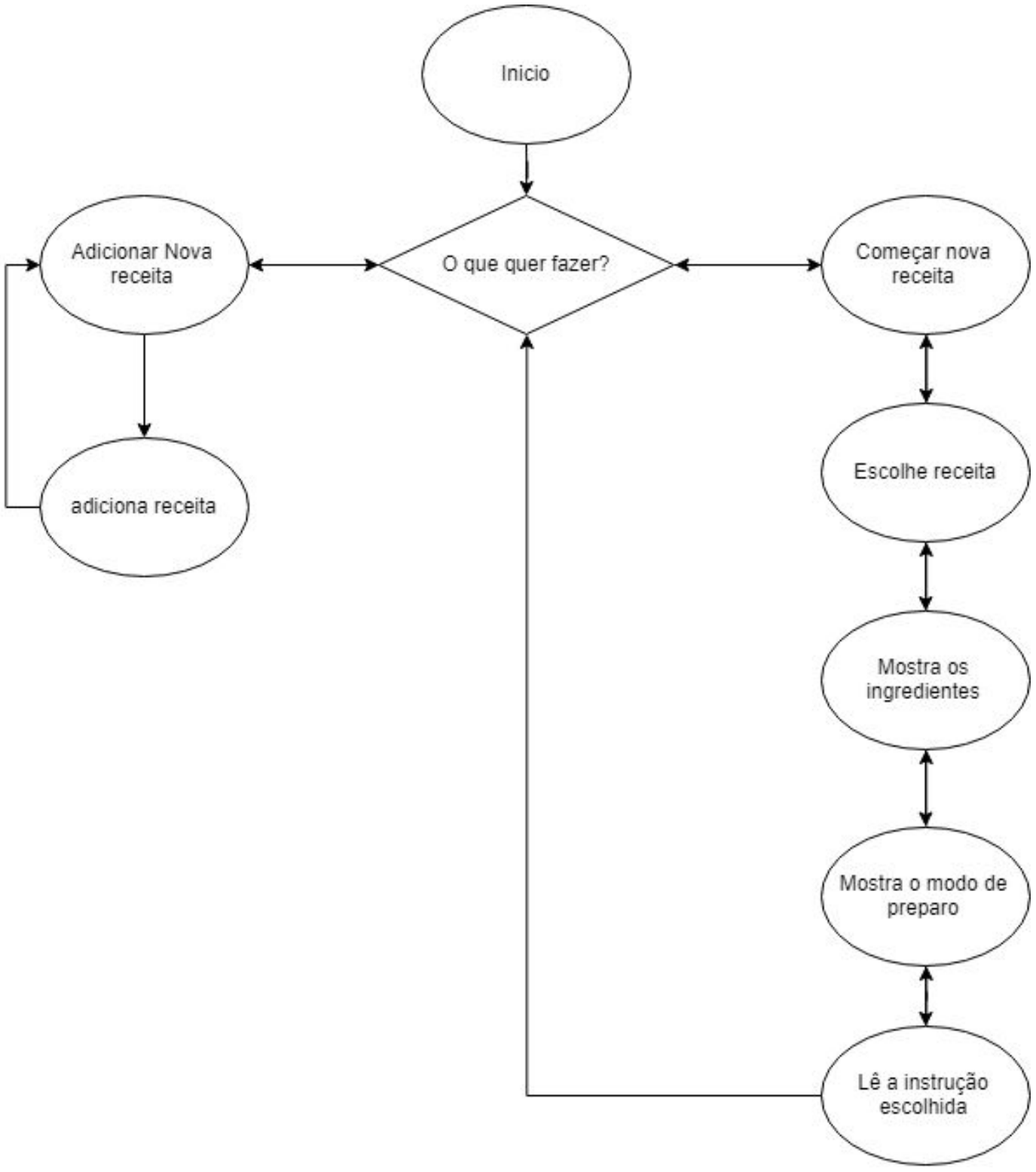
Cozinheiros iniciantes que precisam buscar ajuda constante no modo de preparo durante o preparo da comida, agora podem apenas pedir através de um comando de voz para a raspberry para que ela o diga o que fazer. Para tornar possível este auxílio do computador na cozinha, foi necessário programá-lo em C e C++ para fazer com que comandos fossem executados. O Programa foi desenvolvido e funciona como esperado.

V. Bibliografia

[1]https://kogut.oglobo.globo.com/noticias-da-tv/critica/noticia/2018/08/os-programas-de-culinaria-se-multiplicam-na-tv.html

[2]https://www.raspberrypi.org/products/raspberry-pi-3-model-b/
[3]http://tfalactifry.com/
[4]https://getdrop.com/
[5]https://cloud.google.com/text-to-speech/
[6]https://cloud.google.com/speech-to-text/
[7]https://github.com/goodtft/LCD-show
[8]https://wiki.qt.io/Raspberry_Pi_Beginners_Guide
[9]https://www.mpg123.de/
[10]https://snips.gitbook.io/documentation/advanced-configuration/wakeword/personal-wakeword#5-update-your-assistant-configuration-to-run-your-personal-model

Anexo 1 - Fluxograma do uso do aplicativo para o usuário



Anexo 2 - Telas do programa

Figura 1 - Tela inicial

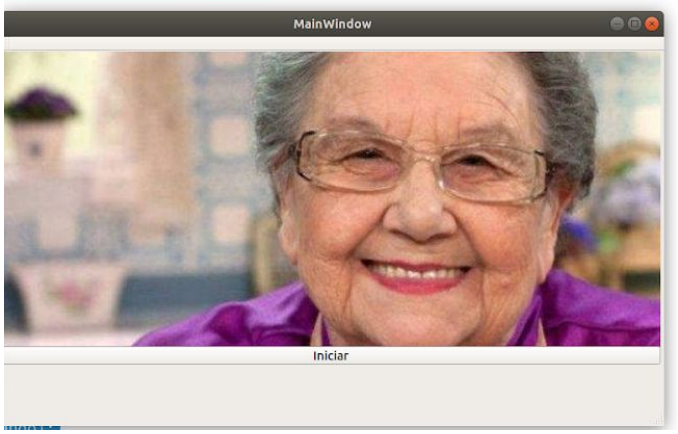


Figura 2 - Tela secundária, escolha entre Cozinhar ou adicionar receita

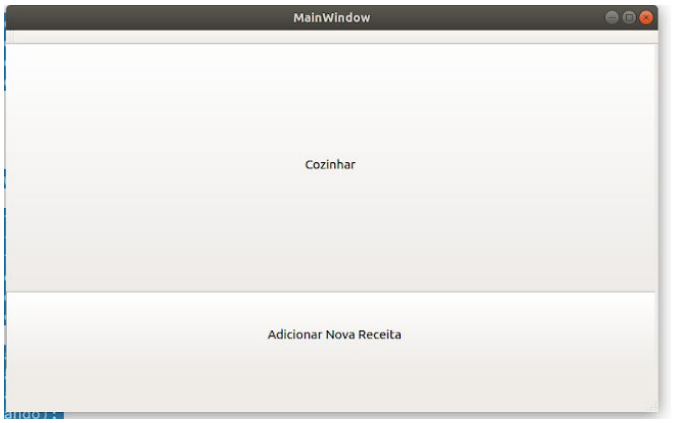


Figura 3 - Adicionar receita

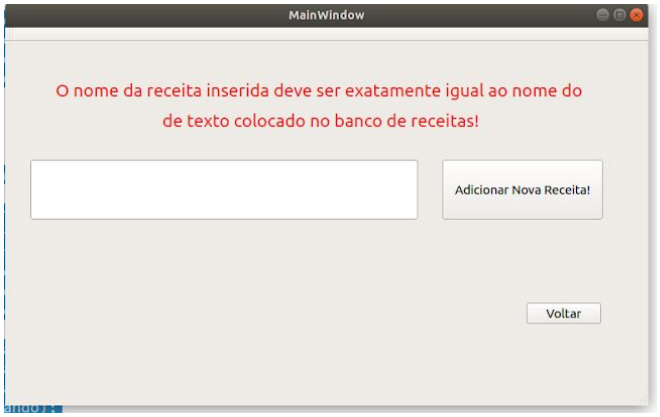


Figura 4 - Primeira tela da opção Cozinhar, mostra opções presentes no banco de receitas.

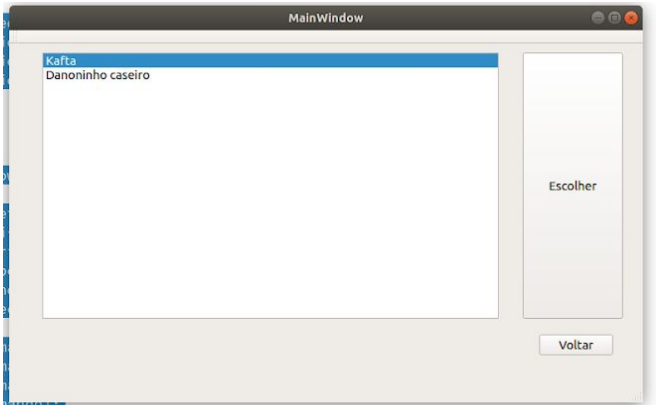


Figura 5 - Tela dos ingredientes da receita escolhida

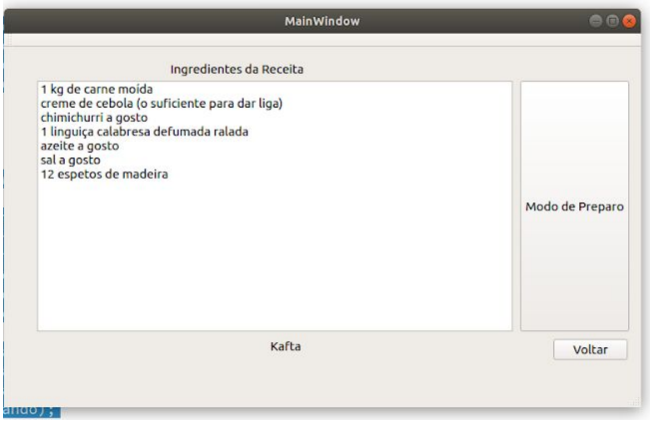
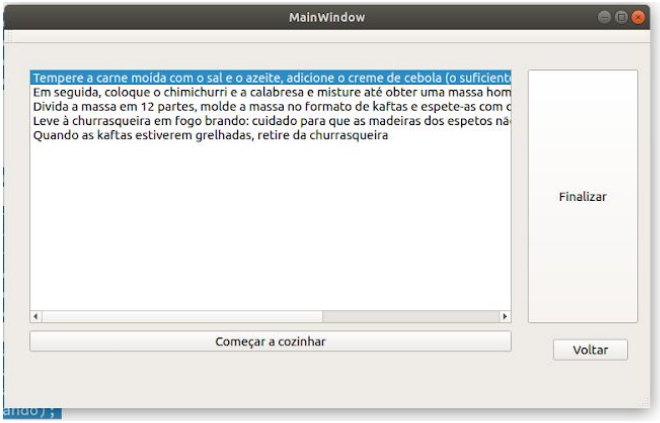
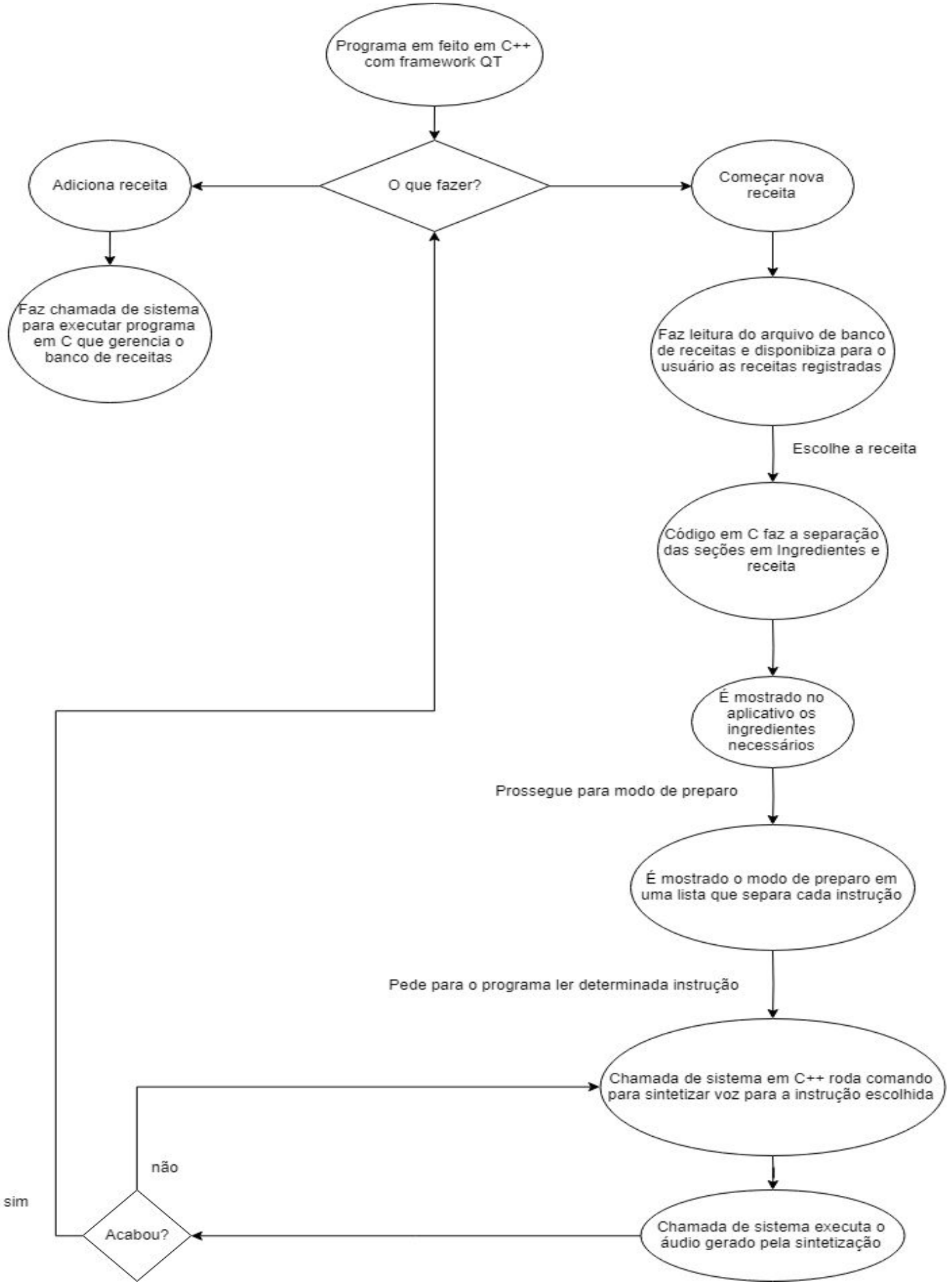


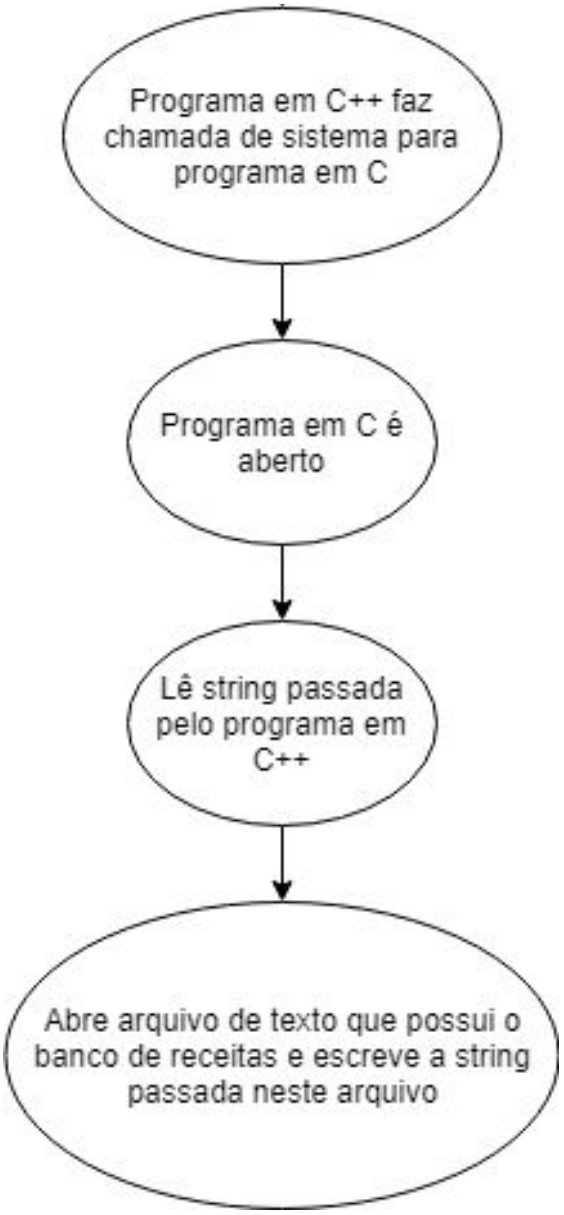
Figura 6 - Tela do modo de preparo



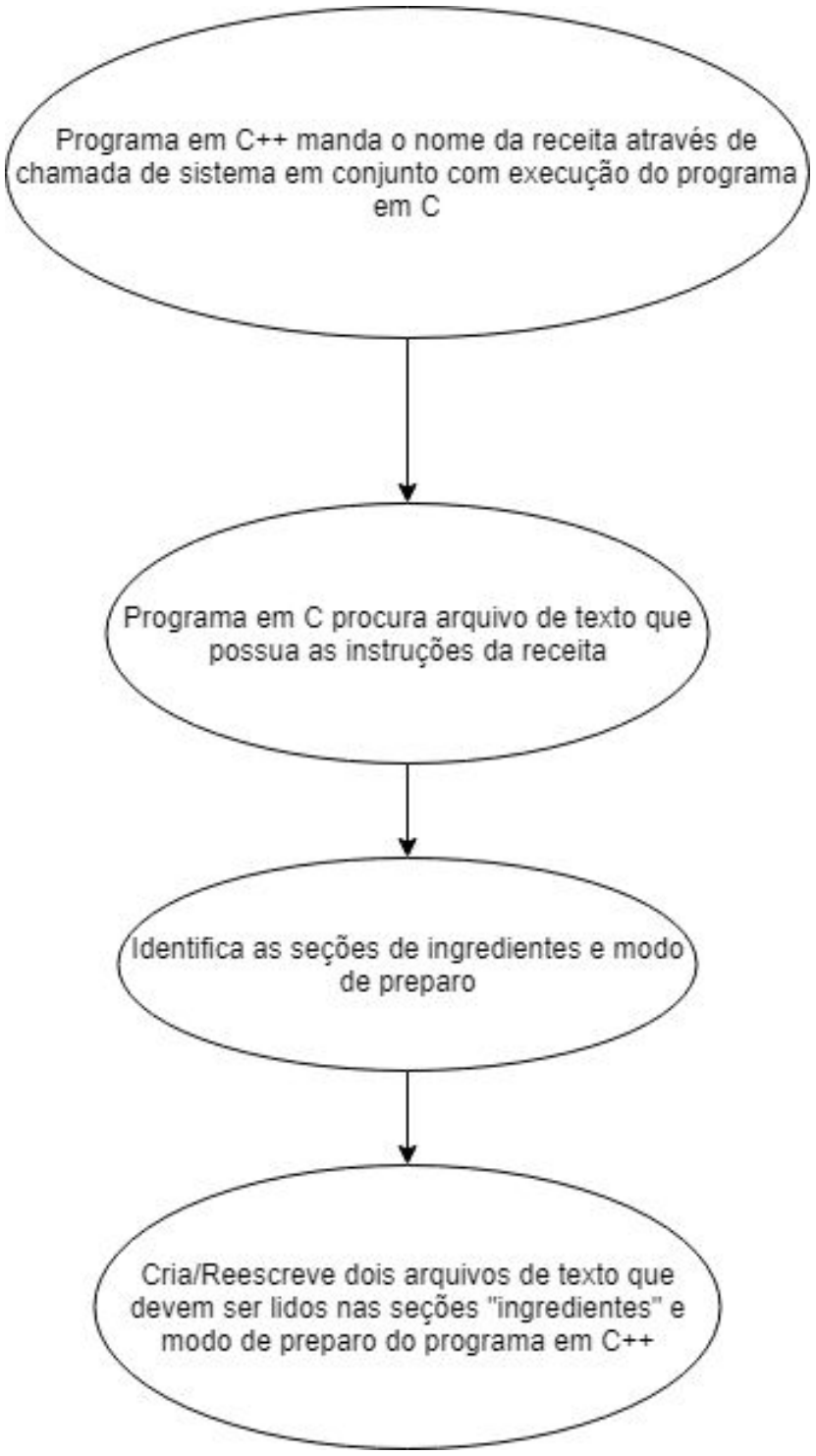
Anexo 3 - Fluxograma do funcionamento do código em C++



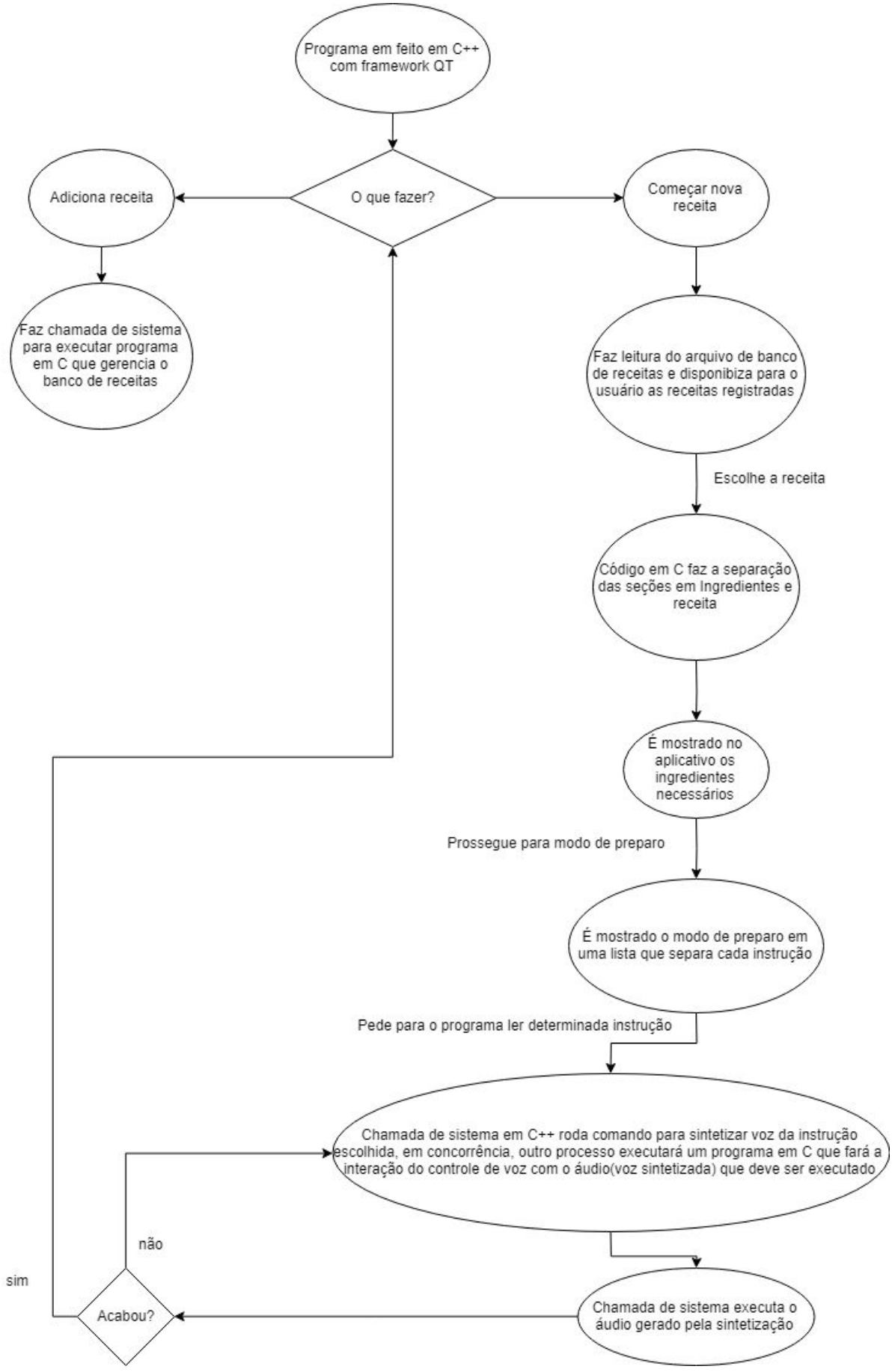
Anexo 4 - Fluxograma do funcionamento banco de receitas



Anexo 5 - Fluxograma funcionamento programa em C que faz separação da receita em ingredientes e modo de preparo



Anexo 6 - Fluxograma do funcionamento pretendido



Anexo 7 - Códigos

Código que separa instruções

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char *argv[ ]){

  FILE *fp;
  int i = 0;
  int armazena;
  int conta = 0;

   char *nome = (char*)malloc(1025*sizeof(char*));

  char **ingredientes = (char**)malloc(1025*sizeof(char*));
   for(int i=0; i<1025; i++){
      ingredientes[i]=(char*)malloc(1025*sizeof(char));
   }

     char **MDP = (char**)malloc(1025*sizeof(char*));
   for(int i=0; i<1025; i++){
      MDP[i]=(char*)malloc(1025*sizeof(char));
  }

   strcpy(nome, "\0");
   for(int o = 1 ; o < argc ;o++){
   if(o == argc){
     break;
   }
   strcat(nome, argv[o]);
   if(o+1 != argc){
   strcat(nome, " ");
  }
  }

   strcat(nome, ".txt");

  printf("%s\n", nome);

   fp = fopen(nome, "r");

  while(fgets(ingredientes[i], 1000, fp) != NULL){

            if(ingredientes[i][15] == ':'){
                      break;
            }

  conta++;

   i++;
  }

    FILE *fpgrava;
    fpgrava = fopen("ingredientes.txt", "w+");


    for(int i = 1; i<conta;i++){
     fputs(ingredientes[i], fpgrava);
    }

    fclose(fpgrava);

    int conta1 = 0;

     char comando[1024];

   while(fgets(MDP[i], 1000, fp) != NULL){
   conta1++;


    i++;
  }

    FILE *fpgrava1;
    fpgrava1 = fopen("mododepreparo.txt", "w+");


    for(int b = conta; b<conta1+conta;b++){
     fputs(MDP[b], fpgrava1);
    }

    fclose(fpgrava1);

          fclose ( fp );

     return 0;
  }
```

Código que grava banco de receitas

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <unistd.h>

void main(int argc, char *argv[ ]) {
  FILE *fp;

     char **ingredientes =
(char**)malloc(1025*sizeof(char*));
```

```
   for(int i=0; i<1025; i++){
      ingredientes[i]=(char*)malloc(1025*sizeof(char));
   }


  fp = fopen("bancodereceitas.txt", "r+");
  int i=0;
  int conta = 0;
 while(fgets(ingredientes[i], 1000, fp) != NULL){

            if(ingredientes[i][15] == ':'){
                        break;
            }

            conta = 0;

  i++;
 }

  for(int o = 1 ; o < argc ;o++){
  fputs(argv[o], fp);
  if(o+1 != argc){
  fputs(" ", fp);
            }
  if(o+1 == argc){
            fputs("\n", fp);
  }
  }
  fclose(fp);
}
```

Arquivo principal do programa que gera a interface gráfica

```
#include "mainwindow.h"
#include <string>
#include "ui_mainwindow.h"
#include "teste.h"
#include <QFile>
#include <QTextStream>


MainWindow::MainWindow(QWidget *parent) :
  QMainWindow(parent),
  ui(new Ui::MainWindow)
{


  ui->setupUi(this);
  ui->stackedWidget->setCurrentIndex(0);
  QPixmap pix("palmirinha.jpg");
  ui->label_3->setPixmap(pix);
//-----------------------------------------------
  int conta = 0;
  QFile inputFile(QString("bancodereceitas.txt"));
  inputFile.open(QIODevice::ReadOnly);
  if (!inputFile.isOpen())
     return;

  QTextStream stream(&inputFile);
  QString line;
  line = stream.readLine();

  conta++;

  while (!line.isNull()) {
     line = stream.readLine();
     conta++;
  }

//----------------------


  QFile inputFile1(QString("bancodereceitas.txt"));
  inputFile1.open(QIODevice::ReadOnly);
  if (!inputFile1.isOpen())
     return;

  QTextStream stream1(&inputFile1);
  QString line1;
  for(int i = 0; i<conta-1;i++){
  line1 = stream1.readLine();
  ui->listWidget->addItem(line1);
  }

//-------------------lê----------------------

}

MainWindow::~MainWindow()
{
  delete ui;
}


void MainWindow::on_pushButton_clicked()
{
  ui->stackedWidget->setCurrentIndex(3);

  ui->listWidget->setCurrentRow(0);

}

void MainWindow::on_stackedWidget_destroyed()
{

}

//----------Volta e reseta
```

```
void MainWindow::on_pushButton_3_clicked()
{
  ui->stackedWidget->setCurrentIndex(1);
  ui->listWidget_2->clear();

}

void MainWindow::on_pushButton_4_clicked()
{
  ui->stackedWidget->setCurrentIndex(2);
  ui->listWidget_2->clear();
}

void MainWindow::on_pushButton_5_clicked()
{

  ui->stackedWidget->setCurrentIndex(3);
  ui->listWidget_2->clear();
  ui->listWidget_3->clear();

}

void MainWindow::on_pushButton_6_clicked()
{
  ui->stackedWidget->setCurrentIndex(1);
  ui->listWidget_2->clear();
}

// BOTA DE ADICIONAR NOVA RECEITA
void MainWindow::on_pushButton_7_clicked()
{
  QString recebe;
  char *recebe1;
  char banco[300];
  recebe = ui->lineEdit->text();

   QByteArray ba = recebe.toLatin1();

  recebe1 = ba.data();
  strcpy(banco, "./gravabanco");
  strcat(banco, " ");
  strcat(banco, recebe1);
  if(recebe1[0] == '\0'){
     ui->label_2->setText("Por favor, escreva o nome da
receita");

  }else{

     system(banco);
     ui->label_2->setText("A Receita foi adicionada!");
     ui->listWidget->addItem(recebe1);

  }
}

// primeiro botao

void MainWindow::on_pushButton_2_clicked()
{
  ui->stackedWidget->setCurrentIndex(1);
  ui->listWidget_2->clear();
}

// BOTAO ESCOLHE A RECEITA

void MainWindow::on_pushButton_8_clicked()
{
  const QString& s = ui->listWidget->currentItem()->text();
  char *recebe1;
  char receita[300];
  ui->label_5->setText(s);
  ui->stackedWidget->setCurrentIndex(4);
  ui->listWidget->setCurrentRow(0);

  QByteArray ba = s.toLatin1();

  recebe1 = ba.data();
  strcpy(receita, "./emba");
  strcat(receita, " ");
  strcat(receita, recebe1);
  system(receita);

  //-------------------------------------------
     int conta = 0;

     QFile inputFile(QString("ingredientes.txt"));
     inputFile.open(QIODevice::ReadOnly);
     if (!inputFile.isOpen())
        return;

     QTextStream stream(&inputFile);
     QString line;
     line = stream.readLine();

     conta++;

     while (!line.isNull()) {
        line = stream.readLine();
        conta++;
     }

  //----------------------


     QFile inputFile1(QString("ingredientes.txt"));
     inputFile1.open(QIODevice::ReadOnly);
     if (!inputFile1.isOpen())
```

```
        return;

    QTextStream stream1(&inputFile1);
    QString line1;

    for(int i = 0; i<conta-1;i++){
    line1 = stream1.readLine();
    ui->listWidget_2->addItem(line1);
    }

  //-------------------lê----------------------
    ui->listWidget->setCurrentRow(0);


}

//---------VAI PRO MODO DE PREPARO
void MainWindow::on_pushButton_9_clicked()
{

  ui->stackedWidget->setCurrentIndex(5);
  //-----------------------------------------------
    int conta = 0;

    QFile inputFile(QString("mododepreparo.txt"));
    inputFile.open(QIODevice::ReadOnly);
    if (!inputFile.isOpen())
        return;

    QTextStream stream(&inputFile);
    QString line;
    line = stream.readLine();

    conta++;

    while (!line.isNull()) {
        line = stream.readLine();
        conta++;
    }

  //----------------------


    QFile inputFile1(QString("mododepreparo.txt"));
    inputFile1.open(QIODevice::ReadOnly);
    if (!inputFile1.isOpen())
        return;

    QTextStream stream1(&inputFile1);
    QString line1;

    char numberstring[2];
    for(int i = 0; i<conta-1;i++){

    line1 = stream1.readLine();
    ui->listWidget_3->addItem(line1);
    char comando[1024];
    strcpy(comando, "gtts-cli \"");
    strcat(comando, line1.toUtf8().constData());
    strcat(comando, "\" --lang pt-br -o ");
    sprintf(numberstring, "%d", i);
    strcat(comando, numberstring);
    strcat(comando, ".mp3");
    system(comando);

    }

  //-------------------lê----------------------
    ui->listWidget_3->setCurrentRow(0);

}

void MainWindow::on_pushButton_10_clicked()
{
  ui->stackedWidget->setCurrentIndex(4);
  ui->listWidget->setCurrentRow(0);


}

void MainWindow::on_pushButton_11_clicked()
{
  ui->stackedWidget->setCurrentIndex(1);
  ui->listWidget_3->clear();
  ui->listWidget_2->clear();
  ui->listWidget->setCurrentRow(0);


}

void MainWindow::on_pushButton_12_clicked()
{
  QListWidgetItem *selected =
ui->listWidget_3->selectedItems().first();
  int q = ui->listWidget_3->row(selected);
  const QString& s = ui->listWidget_3->currentItem()->text();
  char recebe[2];
  char comando[2000];
  sprintf(recebe, "%d", q);

  strcpy(comando, "mpg123 ");
  strcat(comando, recebe);
  strcat(comando, ".mp3");
  system(comando);


}
```