



**CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E  
DESENVOLVIMENTO  
DE SISTEMAS**

**PROJETO INTEGRADOR: DESENVOLVIMENTO DE  
SISTEMA - EAD. MÓDULO - 10**

**CLEVERSON KOZUF**

**CENTRAL TEC.**

**Polo de Mallet |Paraná**

**2022**

**CLEVERSON KOZUF**

**CENTRAL TEC**

Trabalho de Projeto Integrador como conclusão  
de Curso de Graduação em Análise e  
Desenvolvimento de Sistema

Área de concentração:

Orientador: Sirley Ambrosia Vitorio Oliveira

**Polo de Mallet |Paraná**

**2022**

## **RESUMO**

Nos dias atuais é indispensável o uso de tecnologias para auxiliar nos trabalhos diários, diminuindo o tempo gasto com problemas simples que podem ser resolvidos por um programa de computador, ou aplicativo de celular, aumentado assim, a produtividade individual e da empresa. neste projeto será desenvolvido e implementado um programa que será capaz de sanar muitas das dificuldades diárias de uma empresa que trabalha no ramo de manutenção de celulares e computadores, foi feita uma análise detalhada do dia-a-dia da empresa e foi possível detectar os principais problemas enfrentados, no decorrer deste projeto será apresentado várias definições importantes para um bom entendimento das muitas etapas de desenvolvimento de um software. Ao final da leitura será possível entender o funcionamento do programa e fases que compõem uma boa prática de programação, assim como, será apresentado os códigos do seu desenvolvimento, este trabalho é continuação do projeto anterior.

Palavras-chave: Sistemas; Planejamento; Organização; Segurança; Desenvolvimento

## Sumário

|  |           |
|--|-----------|
| <b>1. INTRODUÇÃO .....</b>             | <b>5</b>  |
| 1.1 Apresentação da Empresa.....       | 5         |
| 1.2 Objetivo .....                     | 6         |
| <b>2. METODOLOGIA DA PESQUISA.....</b> | <b>8</b>  |
| <b>3. DESENVOLVIMENTO .....</b>        | <b>8</b>  |
| 3.1 Diagrama UML .....                 | 9         |
| 3.2. Banco de Dados .....              | 10        |
| 3.3. CenTEC.....                       | 15        |
| 3.3. Código do Banco de Dados .....    | 19        |
| 3.4. Código do Software .....          | 20        |
| <b>4. CONCLUSÃO.....</b>               | <b>27</b> |

## 1. INTRODUÇÃO

### 1.1 Apresentação da Empresa

A empresa Central Tec trabalha no ramo de manutenção de celulares e computadores, assim como, venda de acessórios e produtos eletrônicos, com sede única situada no centro da cidade de Mallet-PR, possui somente um funcionário (o desenvolvedor deste projeto), responsável pela manutenção dos aparelhos.

O programa denominado CenTEC visa sanar os problemas da empresa expostos a seguir: Grande dificuldade no gerenciamento de clientes; Apresenta falhas contínuas em seus orçamentos, muitas vezes deixa de repassar ao cliente e perde muitos serviços por conta dessas falhas; Os aparelhos que chegam para manutenção são identificados por etiqueta, que contém os dados do cliente, o valor do orçamento e o problema a ser resolvido, ocorre várias vezes de os funcionários não saberem quem é o dono do celular, ou qual é o defeito do aparelho, por falha na anotação; A empresa também não possui gerenciamento da garantia fornecida por ela em seus produtos e serviços.

Identificando essas falhas podemos encontrar soluções computacionais para as mesmas, a empresa já utilizava um programa de gerenciamento de estoque, mas era muito complexo e pouco otimizado para a loja, contendo poucas funcionalidades utilizadas, além de ter um custo muito elevado, pago na forma de assinatura anual, por estes motivos foi descontinuado o uso do software, voltando a forma tradicional e ultrapassada de gerenciamento.

A seguir será apresentado o protótipo do programa a ser desenvolvido, na sessão Desenvolvimento será mostrado a importância dos diagramas e seu uso, assim como, definições importantes em relação a banco de dados, somente na guia CenTEC será exibido os prints das telas do programa já desenvolvido, explicando as funcionalidades de cada tela, mais adiante será apresentado os códigos de desenvolvimento tanto do programa como do banco de dados.

## 1.2 Objetivo

Um software de controle permite acompanhar as datas de solicitação e entrega dos serviços, o cumprimento de prazos e a qualidade do atendimento prestado pela empresa, é possível registrar as entradas e saídas de smartphones de forma simples e prática.

O software a ser desenvolvido tem como finalidade ajudar no gerenciamento dos clientes, controlar os aparelhos que estão na loja para serem consertados, assim como, o prazo de entrega e tempo de garantia decorrido. O protótipo do programa é apresentado nas **figuras (1.1, 1.2 ,1.3, 1.4, 1.5 e 1.6).**

A tela inicial do sistema apresenta um layout com botões de navegação e uma tabela de dados. Os botões são: 'ADD CLIENTE', 'ENTRADA E SAÍDA DE APARELHOS', 'CONSULTAR DB', 'APARELHOS', 'GARANTIA', 'NOME DO CLIENTE' (com campo de texto), 'BUSCAR' e 'LIMPAR'. A tabela abaixo contém uma única linha de dados.

| ID | NOME              | TELEFONE    | STATUS | ID Aparelho |
|----|-------------------|-------------|--------|-------------|
| 1  | Matheus Guilherme | 42998665859 | Pronto | 42998665859 |
|    |                   |             |        |             |
|    |                   |             |        |             |
|    |                   |             |        |             |
|    |                   |             |        |             |

Figura 1.1<tela inicial> Autor <cleverson kozuf>

A tela para adicionar novos clientes possui campos de entrada para 'Nome', 'Telefone' e 'CPF do cliente'. Cada campo é precedido por um rótulo. À direita dos campos de 'Telefone' e 'CPF do cliente' há um botão 'LIMPAR'. Na parte inferior da tela, há dois botões: 'ADD Cliente' e 'ADD aparelho'.

Figura 1.2<tela adicionar clientes> Autor <cleverson kozuf>

[illegible]

Marca

Modelo

Orçamento

Descrição

ID Aparelho

ID Cliente

ID Funcionário

ADD

ID Aparelho

DAR BAIXA





### 3.1 Diagrama UML

A Linguagem Unificada de Modelagem, é, como o nome indica, uma linguagem de notação utilizada para modelar e documentar as diversas fases do desenvolvimento de sistemas orientados a objetos. Para isso, ela define uma série de elementos gráficos (como retângulos, setas, balões e linhas) que são usados em diferentes diagramas para representar os componentes de uma aplicação, suas interações e mudanças de estados. Trata-se de uma linguagem de modelagem única, cujo papel é auxiliar a equipe de desenvolvimento a visualizar os diversos aspectos da aplicação, facilitando a compreensão do seu funcionamento.

É uma linguagem apoiada por um metamodelo único, que ajuda na descrição e no projeto de sistemas de software, particularmente daqueles construídos utilizando o estilo orientado a objetos (FOWLER et al., 2005)

São essenciais para evitar dois problemas comuns no desenvolvimento de software: os erros das fases de especificação do projeto e a comunicação entre as diferentes partes envolvidas, como gerentes, pessoas desenvolvedoras, analistas, por exemplo.

A Linguagem UML inclui dez diagramas diferentes para representar os aspectos estruturais, comportamentais e físicos de um software (BOOCH et al., 2000).

Com o uso desses diagramas, é possível obter uma visão clara e única do sistema, deixando todas as entidades envolvidas no projeto conscientes do que será desenvolvido e evitando erros de implementação. Afinal, trata-se de uma linguagem padrão, objetiva e eficiente que facilmente será entendida por toda a equipe. Um diagrama muito importante utilizado para o desenvolvimento de software, é o modelo entidade-relacionamento (MER) representado na **Figura 2.1**.

### 3.2. Banco de Dados

Simplificadamente, a função de um banco de dados é coletar e armazenar informações desejadas. Assim, permitindo que os dados sejam interpretados e analisados facilmente. Dessa forma, por exemplo, uma empresa pode acessar seu banco de dados e verificar quais produtos estão sendo vendidos e organizá-los por preço, categoria ou data de venda. Ou seja, os bancos de dados, além de necessários, são extremamente cômodos no que diz respeito ao esforço. Pois, torna-se muito fácil a coleta e a leitura de dados se compararmos com os métodos manuais, como anotações em papel e planilhas, e isso potencializa os negócios.

A expressão Banco de Dados originou-se do termo inglês Databanks. Este foi trocado pela palavra Databases (Base de Dados) devido possuir significação mais apropriada (SETZER, 2005).

Para Date (2004), um banco de dados é uma coleção de dados persistentes, usadas pelos sistemas de aplicação de uma determinada empresa.

#### **TIPOS DE BANCO DE DADOS**

Existem dois principais tipos de BANCO DE DADOS que as empresas utilizam, o primeiro e mais comum no mercado é o relacional. Esses são principalmente encontrados em sistemas ERP e CRM, sua popularidade se deve ao fato da facilidade da armazenagem e pela confiabilidade das informações. Além disso, seu funcionamento é baseado no modelo SQL, de forma que a inserção dos dados é fácil e recuperável, e seu armazenamento é construído em formatos tabulares. Resumidamente, SQL é uma linguagem de consulta estruturada direcionada a lidar com bancos de dados armazenados em tabelas.

Por outro lado, o segundo tipo é conhecido como não relacional e é geralmente utilizado ao trabalhar com dados que não podem ser inseridos em formato tabela, como imagens, vídeos e gráficos. Seu destaque é na sua alta performance, por isso é um banco de dados muito valorizado no mercado. Logo, por ser um sistema que armazena informações de maior complexidade, sua linguagem é NoSQL. Ou seja, esse modelo não possui uma linguagem de consulta estruturada em tabelas simples como visto no relacional, embora mais

complicado, sua principal vantagem é permitir identificar qual tipo de informação é mais relevante para o negócio, o que é um grande benefício, é amplamente utilizado por redes sociais.

De acordo com DATE (2004, p. 6), um sistema de banco de dados é “um sistema computadorizado cuja finalidade geral é armazenar informações e permitir que os usuários busquem e atualizem essas informações quando as solicitar”.

Dada as definições dos tipos de banco de dados, é possível concluir que o modelo ideal para nosso programa é o banco de dados relacional SQL, as imagens da criação do banco de dados “DB\_CENTRAL” na linguagem SQL são apresentadas na subseção **3.3 (Código do Banco de Dados)**. Para sua criação foi de suma importância os diagramas de caso de uso (**Figura 2.3**), entidade-relacionamento (**Figura 2.1**) e o fluxograma (**Figura 2.4**). Depois de concluir a análise desses diagramas, foi possível criar a tabela representada na (**figura 2.2**), que contém para cada entidade os atributos simples e as suas chaves primárias (PK) e as chaves secundárias, ou estrangeiras (FK), auxiliando muito na hora da codificação do banco de dados.

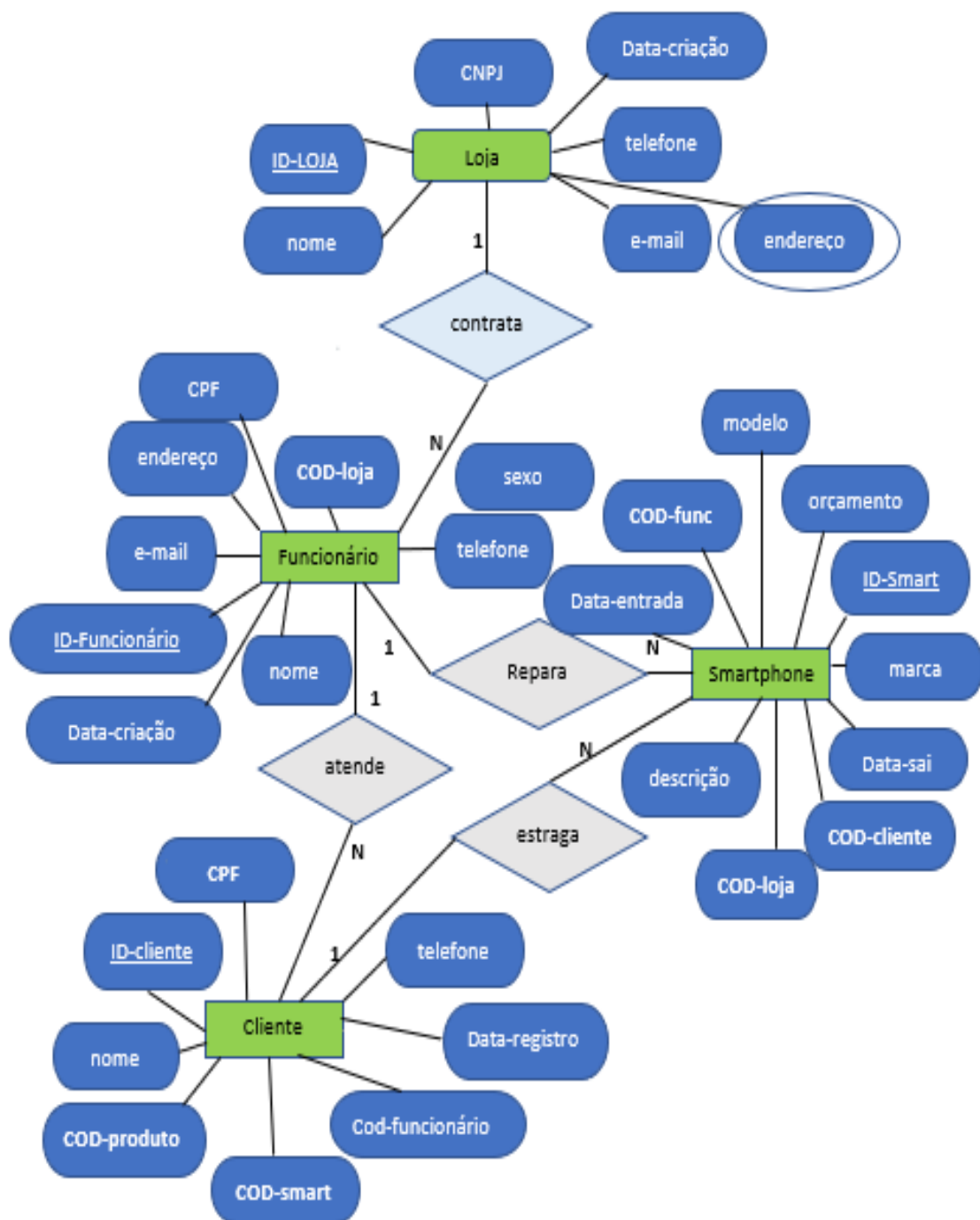


Figura 2.1 <diagrama entidade relacionamento> Autor <cleverson kozuf>

**Loja**

| <u>ID-LOJA</u> | Nome     | Telefone | CNPJ     | E-mail   | Endereço | Data-criação |
|----------------|----------|----------|----------|----------|----------|--------------|
| Pk             | Varchar  | Numeric  | Numeric  | Varchar  | Varchar  | Datetime     |
| Not null       | Not null | Not null | Not null | Not null | Not null | Auto         |

**Funcionário**

| <u>ID-FUNCIONÁRIO</u> | Nome     | CPF     | Telefone | E-mail   | Endereço | Data-criação | Sexo    | Cod-loja |
|-----------------------|----------|---------|----------|----------|----------|--------------|---------|----------|
| Pk                    | Varchar  | Numeric | Numeric  | Varchar  | Varchar  | Datetime     | Varchar | fk       |
| Not null              | Not null |         | Not null | Not null | Not null | Auto         |         |          |

**Smartphone**

| <u>ID-SMART</u> | Marca   | Modelo   | Orçamento | Data-entrada | Data-saída | Descrição | COD-loja | COD-cliente | COD-funcionário |
|-----------------|---------|----------|-----------|--------------|------------|-----------|----------|-------------|-----------------|
| Pk              | Varchar | Varchar  | numeric   | Datetime     | Datetime   | Varchar   | fk       | fk          | Fk              |
| Not null        |         | Not null |           | Not null     | Auto       | Not null  |          |             |                 |

**Cliente**

| <u>ID-CLIENTE</u> | Nome     | Telefone | CPF     | Data-registro | COD-smart | COD-funcionário |
|-------------------|----------|----------|---------|---------------|-----------|-----------------|
| Pk not            | Varchar  | Varchar  | numeric | Datetime      | fk        | fk              |
| Not null          | Not null | Not null |         | Auto          |           |                 |

Figura 2.2<tabela banco de dados> Autor <cleverson kozuf>

Para a criação do diagrama (MER) foi muito relevante o auxílio da notação BPMN que é extremamente útil para descrever passo a passo a lógica do processo por meio de diagramas. A partir dessa modelagem, é possível ter uma visão gráfica que expressa de maneira simples e direta todo o processo de funcionamento do negócio (**Figura 2.4**), com ela é possível entender melhor o fluxo de trabalho desenvolvido pela empresa e identificar suas entidades e seus respectivos relacionamentos. Foi criado também um diagrama de caso de uso nível 1 representado na **Figura 2.3**.

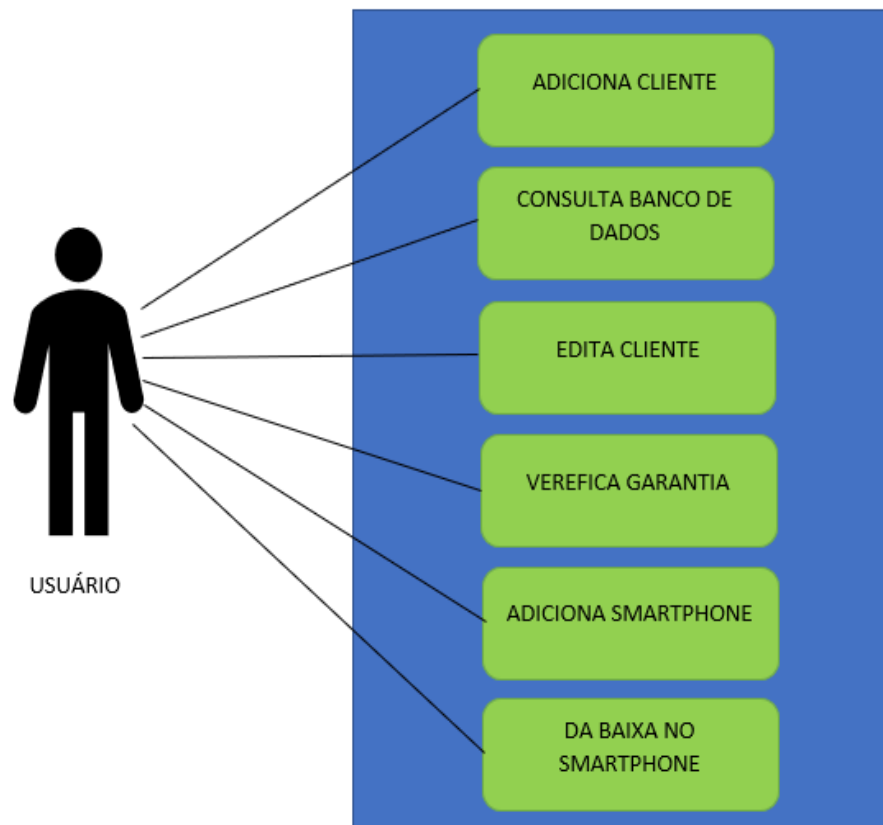


Figura 2.3 <diagrama caso de uso> Autor <cleverson kozuf>

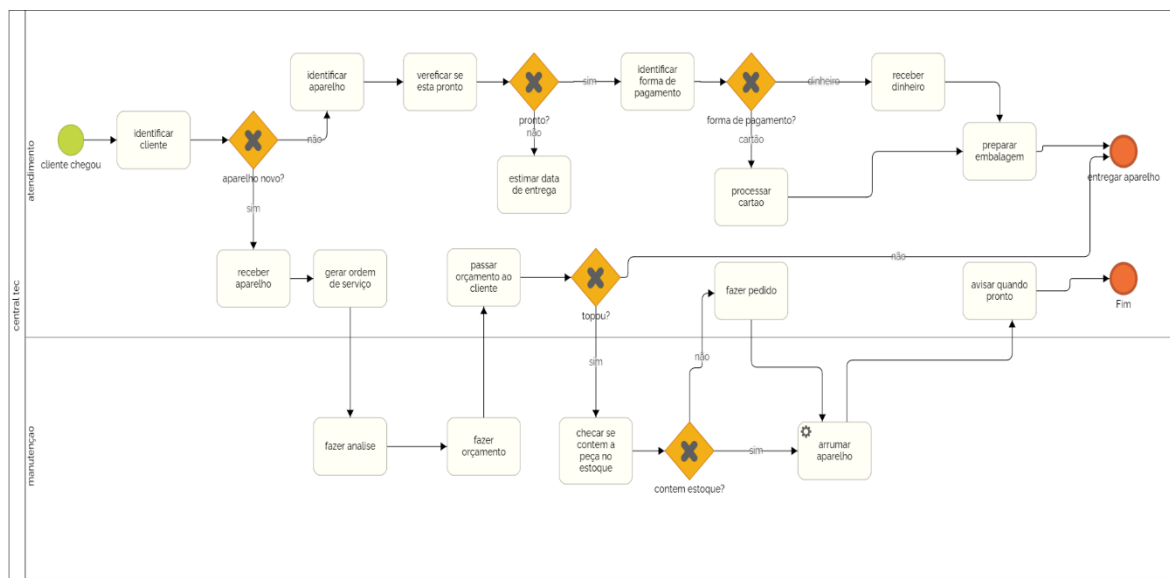


Figura 2.4 <fluxograma> Autor <cleverson kozuf>

### 3.3. CenTEC

A seguir será apresentado o programa “CenTEC” já em funcionamento, suas principais características e funcionalidades:

Tela **INICIAL**, botões que dão acesso a outras telas (ADD CLIENTES, COLSULTAR DB, APARELHOS, GARANTIA, ENTRADA E SAÍDA DE APARELHOS), um campo de busca para agilizar a consulta de determinado cliente, que exibe suas informações, na parte inferior é mostrado uma prévia dos clientes cadastrados e o resultado da busca efetuada. **figura 3.1.**



Figura 3.1< Tela inicial> Autor <cleverson kozuf>

Tela **ADD CLIENTES**; responsável pelo cadastro de clientes, contém os campos (NOME, TELEFONE, CPF DO CLIENTE), com o botão LIMPAR é possível limpar as informações digitadas antes do cadastro definitivo, clicando no botão ADD CLIENTE é efetuado o cadastro definitivo no banco de dados, e o botão ADD APARELHO encaminha o usuário até a tela (ENTRADA E SAÍDA DE APARELHOS). **Figura 3.2.**

Nome

Telefone  LIMPAR

CPF do cliente

ADD Cliente

ADD aparelho

Figura 3.2< Tela ADD clientes > Autor <cleverson kozuf>

Tela GARANTIA; Acompanhamento da garantia, exibe os aparelhos que já foram entregues aos seus clientes e suas informações relevantes, bem como a data de término da garantia, facilitando assim a verificação caso algum cliente acione a garantia. **Figura 3.3.**

| ID-Aparelho | Nome                    | Telefone    | Data-Registro |
|-------------|-------------------------|-------------|---------------|
| 7           | bernadete               | 4299973456  | 20/05/22      |
| 12          | celia                   | 1           | 20/05/22      |
| 2           | cleverson kozuf         | 42998661226 | 20/05/22      |
| 10          | gabriel                 | 1           | 20/05/22      |
| 4           | Isaura nunes            | 42998124854 | 20/05/22      |
| 6           | jaqueline shimik        | 42999348166 | 20/05/22      |
| 13          | marcelo bach            | 42999709066 | 20/05/22      |
| 8           | marcos                  | 42988320031 | 20/05/22      |
| 1           | matheus renan grabowski | 42998603381 | 20/05/22      |
| 15          | teste                   | 42          | 20/05/22      |
| 14          | teste fk                | 1           | 20/05/22      |
| 5           | valdevino               | 984163108   | 20/05/22      |
| 11          | vera ivaninski          | 1           | 20/05/22      |

Figura 3.3< Tela garantia> Autor <cleverson kozuf>



Tela **CONSULTAR DB**; Consulta ao banco de dados, exibe todos os clientes e suas respectivas informações, possui três botões na parte superior da tela (EXCLUIR, EDITAR, ADD), são de fácil acesso e autoexplicativos. **Figura 3.4.**



| id | nome                    | telefone    | Data de registro |
|----|-------------------------|-------------|------------------|
| 7  | bernadete               | 4299973456  | 20/05/22         |
| 12 | celia                   | 1           | 20/05/22         |
| 2  | cleverson kozuf         | 42998661226 | 20/05/22         |
| 10 | gabriel                 | 1           | 20/05/22         |
| 4  | Isaura nunes            | 42998124854 | 20/05/22         |
| 6  | jaqueline shimik        | 42999348166 | 20/05/22         |
| 13 | marcelo bach            | 42999709066 | 20/05/22         |
| 8  | marcos                  | 42988320031 | 20/05/22         |
| 1  | matheus renan grabowski | 42998603381 | 20/05/22         |
| 15 | teste                   | 42          | 20/05/22         |
| 14 | teste fk                | 1           | 20/05/22         |
| 5  | valdevino               | 984163108   | 20/05/22         |
| 11 | vera ivaninski          | 1           | 20/05/22         |

Figura 3.4 < Tela consultar DB> Autor <cleverson kozuf>

Tela **ENTRADA E SAÍDA DE APARELHOS**; responsável pelo cadastro de aparelhos para manutenção, possui quatro campos para serem preenchidos pelo usuário (marca, modelo, orçamento e descrição), no banco de dados é preenchido a data e hora automaticamente. O programa irá gerar o ID do aparelho, o ID do cliente será preenchida pelo usuário, assim como o ID do usuário para ter controle de qual usuário recebeu o celular. Na parte inferior terá um campo para dar baixa nos aparelhos entregues, é só preencher o campo (ID do aparelho) e acionar o botão (DAR BAIXA), o programa irá atualizar o banco de dados e dar baixa no aparelho, que será mostrado a partir desta data com contagem regressiva de 90 dias na tela de GARANTIA. **Figura 3.5.**

Figura 3.5< Tela e entrada e saída de aparelhos > Autor <cleverson kozuf>

Tela **APARELHOS**; controle dos aparelhos que estão na loja aguardando o conserto ou a análise para o orçamento. **Figura 3.6.**

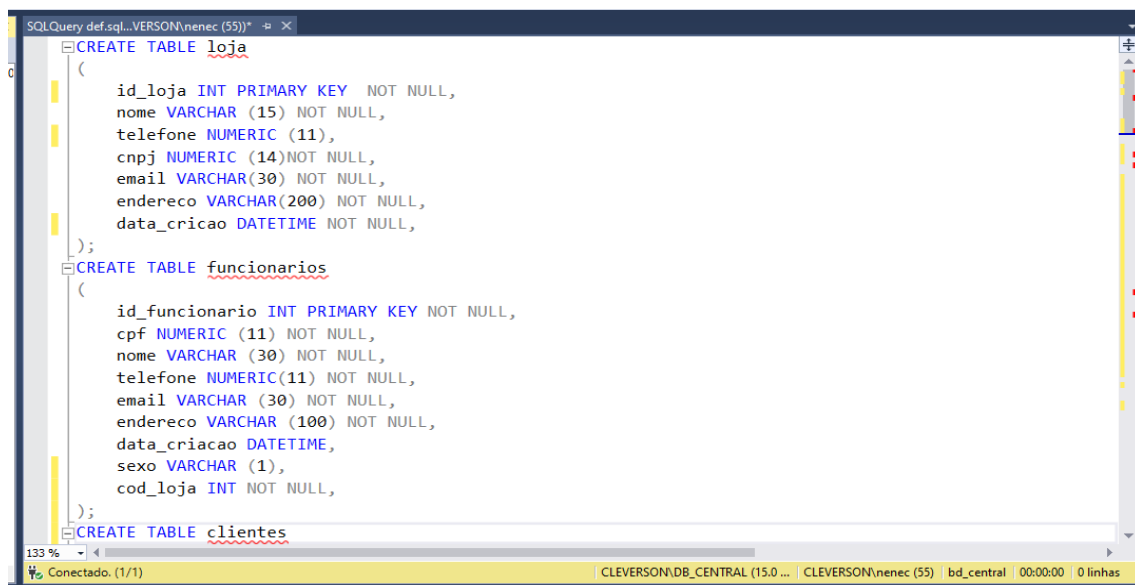
| id | cod | MARCA    | MODELO     | ORÇAMENTO | DATA_ENTRA | DESCRIÇÃO                  | STATUS |
|----|-----|----------|------------|-----------|------------|----------------------------|--------|
| 1  | 1   | xiaomi   | redmi9     | 150.00    | 20/05/22   | troca do conector          | None   |
| 2  | 2   | xiaomi   | redmi 9    | 230.00    | 20/05/22   | tela trincada              | None   |
| 6  | 5   | samsung  | sm-t111m   | 470.00    | 20/05/22   | troca de tela sem garantia | None   |
| 7  | 6   | motorola | x3         | 360.00    | 20/05/22   | troca dela                 | None   |
| 8  | 0   | motorola | xt-1723    | 250.00    | 20/05/22   | troca de tela              | None   |
| 9  | 7   | motorola | xt-1723    | 250.00    | 20/05/22   | troca de tela              | None   |
| 10 | 8   | motorola | xt-1925-3  | 350.00    | 20/05/22   | placa oxidada sem conserto | None   |
| 11 | 10  | motorola | xt-1965-2  | 310.00    | 20/05/22   | tela- sem garantia         | None   |
| 12 | 11  | samsung  | a037m/ds   | 410.00    | 20/05/22   | com garantia               | None   |
| 13 | 12  | motorola | xt-1802    | 295.00    | 20/05/22   | troca de tela              | None   |
| 14 | 13  | samsung  | sm-530h/ds | 240.00    | 20/05/22   | troca de tela              | None   |
| 15 | 14  | teste    | teste      | 150.00    | 20/05/22   | teste                      | None   |
| 17 | 14  | teste    | test       | 150.00    | 20/05/22   | test                       | None   |
| 18 | 14  | de       | de         | 15.00     | 20/05/22   | teste                      | None   |

Figura 3.6< Tela aparelhos > Autor <cleverson kozuf>

Como está sendo desenvolvido um programa para uso real, futuramente contará com novas funcionalidades como; controle de estoque de películas e capinhas, controle de peças utilizadas em seus respectivos aparelhos, para saber a origem da peça e para qual fornecedor deverá ser encaminhada em caso

de acionamento da garantia, controle de clientes inadimplentes, além das melhorias visuais e de layout. A seguir será apresentado o código utilizado para criar o banco de dados e o código de desenvolvimento do programa CenTEC.

### 3.3. Código do Banco de Dados



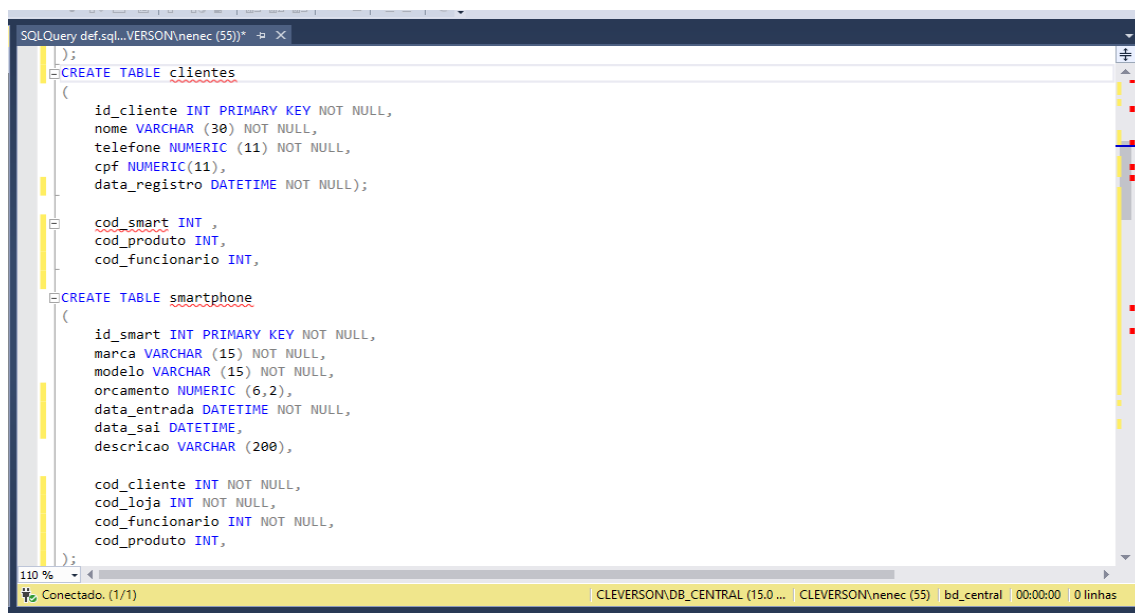
```

SQLQuery def.sql...VERSON\nenec (55)*
CREATE TABLE loja
(
    id_loja INT PRIMARY KEY NOT NULL,
    nome VARCHAR (15) NOT NULL,
    telefone NUMERIC (11),
    cnpj NUMERIC (14) NOT NULL,
    email VARCHAR(30) NOT NULL,
    endereco VARCHAR(200) NOT NULL,
    data_criacao DATETIME NOT NULL,
);
CREATE TABLE funcionarios
(
    id_funcionario INT PRIMARY KEY NOT NULL,
    cpf NUMERIC (11) NOT NULL,
    nome VARCHAR (30) NOT NULL,
    telefone NUMERIC(11) NOT NULL,
    email VARCHAR (30) NOT NULL,
    endereco VARCHAR (100) NOT NULL,
    data_criacao DATETIME,
    sexo VARCHAR (1),
    cod_loja INT NOT NULL,
);
CREATE TABLE clientes

```

133 %  
Conectado. (1/1) CLEVERSON\DB\_CENTRAL (15.0 ... CLEVERSON\nenec (55) bd\_central 00:00:00 0 linhas

Figura 3.1 <criação das tabelas loja e funcionários> Autor <cleverson kozuf>



```

SQLQuery def.sql...VERSON\nenec (55)*
);
CREATE TABLE clientes
(
    id_cliente INT PRIMARY KEY NOT NULL,
    nome VARCHAR (30) NOT NULL,
    telefone NUMERIC (11) NOT NULL,
    cpf NUMERIC(11),
    data_registro DATETIME NOT NULL);

    cod_smart INT ,
    cod_produto INT,
    cod_funcionario INT,
CREATE TABLE smartphone
(
    id_smart INT PRIMARY KEY NOT NULL,
    marca VARCHAR (15) NOT NULL,
    modelo VARCHAR (15) NOT NULL,
    orcamento NUMERIC (6,2),
    data_entrada DATETIME NOT NULL,
    data_sai DATETIME,
    descricao VARCHAR (200),

    cod_cliente INT NOT NULL,
    cod_loja INT NOT NULL,
    cod_funcionario INT NOT NULL,
    cod_produto INT,
);

```

110 %  
Conectado. (1/1) CLEVERSON\DB\_CENTRAL (15.0 ... CLEVERSON\nenec (55) bd\_central 00:00:00 0 linhas

Figura 3.2 <criação das tabelas clientes e smartphone> Autor <cleverson kozuf>

```

SQLQuery def.sql...VERSION\nenec (55))* X
);
--ALTER para adicionar fk nas tabelas

ALTER TABLE funcionarios ADD CONSTRAINT fk_cod_loja FOREIGN KEY (cod_loja) REFERENCES loja (id_loja);
ALTER TABLE clientes ADD CONSTRAINT fk_cod_smart FOREIGN KEY (cod_smart) REFERENCES smartphone (id_smart);
ALTER TABLE clientes ADD CONSTRAINT fk_cod_produto FOREIGN KEY (cod_produto) REFERENCES produtos (id_produto);
ALTER TABLE clientes ADD CONSTRAINT fk_cod_funcionario FOREIGN KEY (cod_funcionario) REFERENCES funcionarios (id_funcionario);
ALTER TABLE smartphone ADD CONSTRAINT fk_cod_cliente FOREIGN KEY (cod_cliente) REFERENCES clientes (id_cliente);
ALTER TABLE smartphone ADD CONSTRAINT fk_cod_loja FOREIGN KEY (cod_loja) REFERENCES loja (id_loja);
ALTER TABLE smartphone ADD CONSTRAINT fk_cod_func FOREIGN KEY (cod_funcionario) REFERENCES funcionarios (id_funcionario);
ALTER TABLE smartphone ADD CONSTRAINT fk_cod_prod FOREIGN KEY (cod_produto) REFERENCES produtos (id_produto);

```

Figura 3.3 <alterações nas tabelas para adição das chaves estrangeiras> Autor <cleverson kozuf>

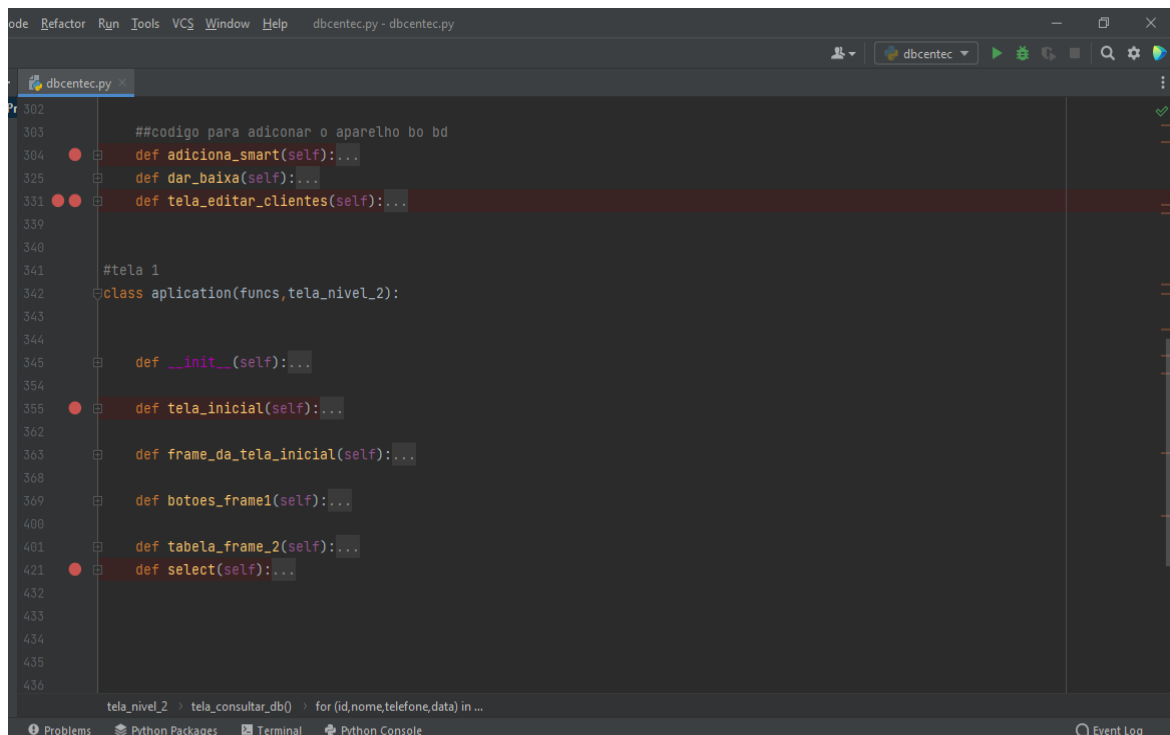
### 3.4. Código do Software

```

dbcentec.py
1  import pyodbc
2  from tkinter import *
3  from tkinter import ttk
4  import awesometkinter as atk
5
6
7
8  root = Tk()
9  #conecta bd
10 class funcs():
11     def limpa_tela(self):...
12     def limpa_tela_inicial(self):
13         self.entrada_nome_i.delete(0, END)
14     def conecta_bd(self):...
15     def desconecta_bd(self):...
16     #tela 2
17 class tela_nivel_2():
18     def tela_add_clientes(self):...
19     def adicione_cliente(self):...
20     def tela_consultar_db(self):...
21     def tela_garantia(self):...
22
23     def tela_entrada_saida_aparelhos(self):...
24     def tela_aparelhos(self):...
25
26     ##codigo para adicionar o aparelho bo bd
27     tela_nivel_2 > tela_consultar_db() > for (id,nome,telefone,data) in ...

```

Figura 4.0 < Todas funções > Autor <cleverson kozuf>

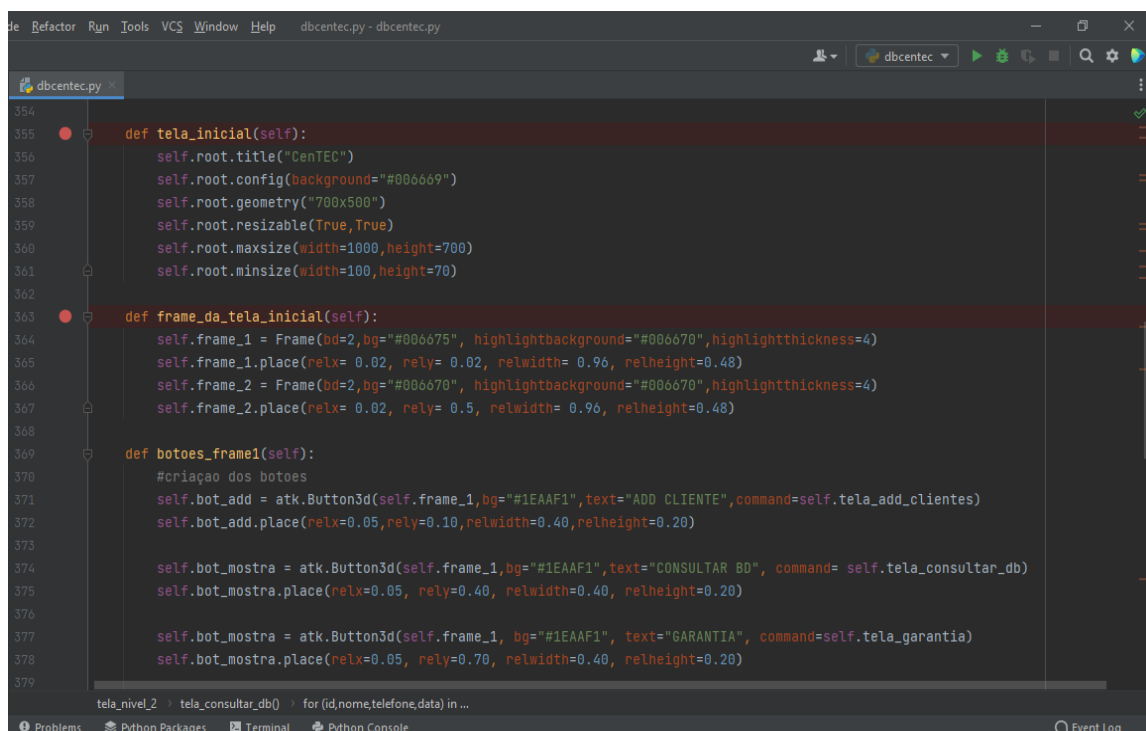


```

302
303     ##codigo para adicionar o aparelho bo bd
304     def adiciona_smart(self):...
305     def dar_baixa(self):...
306     def tela_editar_clientes(self):...
307
308
309
310
311
312
313
314     #tela 1
315     class aplicacion(funcs,tela_nivel_2):
316
317
318     def __init__(self):...
319
320     def tela_inicial(self):...
321
322     def frame_da_tela_inicial(self):...
323
324     def botoes_frame1(self):...
325
326     def tabela_frame_2(self):...
327     def select(self):...
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Figura 4.1 < Todas funções > Autor <cleverson kozuf>

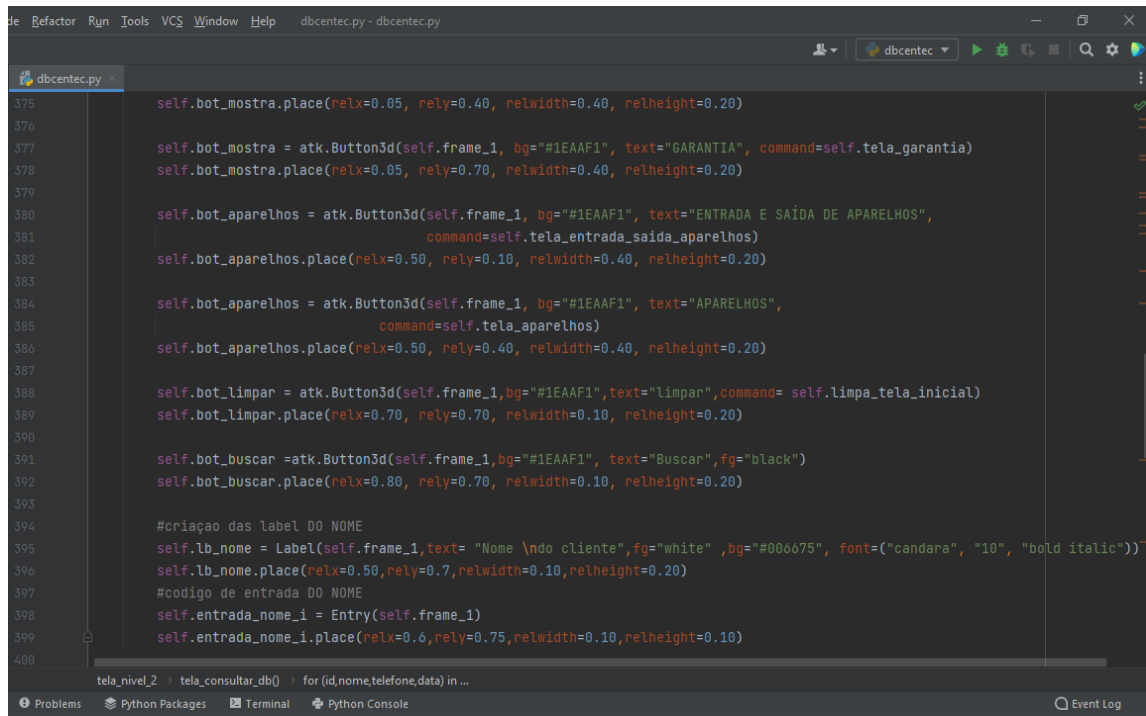


```

354
355     def tela_inicial(self):
356         self.root.title("CenTEC")
357         self.root.config(background="#006669")
358         self.root.geometry("700x500")
359         self.root.resizable(True, True)
360         self.root.maxsize(width=1000,height=700)
361         self.root.minsize(width=100,height=70)
362
363     def frame_da_tela_inicial(self):
364         self.frame_1 = Frame(bd=2,bg="#006675", highlightbackground="#006670",highlightthickness=4)
365         self.frame_1.place(relx= 0.02, rely= 0.02, relwidth= 0.96, relheight=0.48)
366         self.frame_2 = Frame(bd=2,bg="#006670", highlightbackground="#006670",highlightthickness=4)
367         self.frame_2.place(relx= 0.02, rely= 0.5, relwidth= 0.96, relheight=0.48)
368
369     def botoes_frame1(self):
370         #criação dos botoes
371         self.bot_add = atk.Button3d(self.frame_1,bg="#1EAAF1",text="ADD CLIENTE",command=self.tela_add_clientes)
372         self.bot_add.place(relx=0.05,relx=0.10,relwidth=0.40,relheight=0.20)
373
374         self.bot_mostra = atk.Button3d(self.frame_1,bg="#1EAAF1",text="CONSULTAR BD", command= self.tela_consultar_db)
375         self.bot_mostra.place(relx=0.05, rely=0.40, relwidth=0.40, relheight=0.20)
376
377         self.bot_garantia = atk.Button3d(self.frame_1, bg="#1EAAF1", text="GARANTIA", command=self.tela_garantia)
378         self.bot_garantia.place(relx=0.05, rely=0.70, relwidth=0.40, relheight=0.20)
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Figura 4.2 < Tela inicial > Autor <cleverson kozuf>

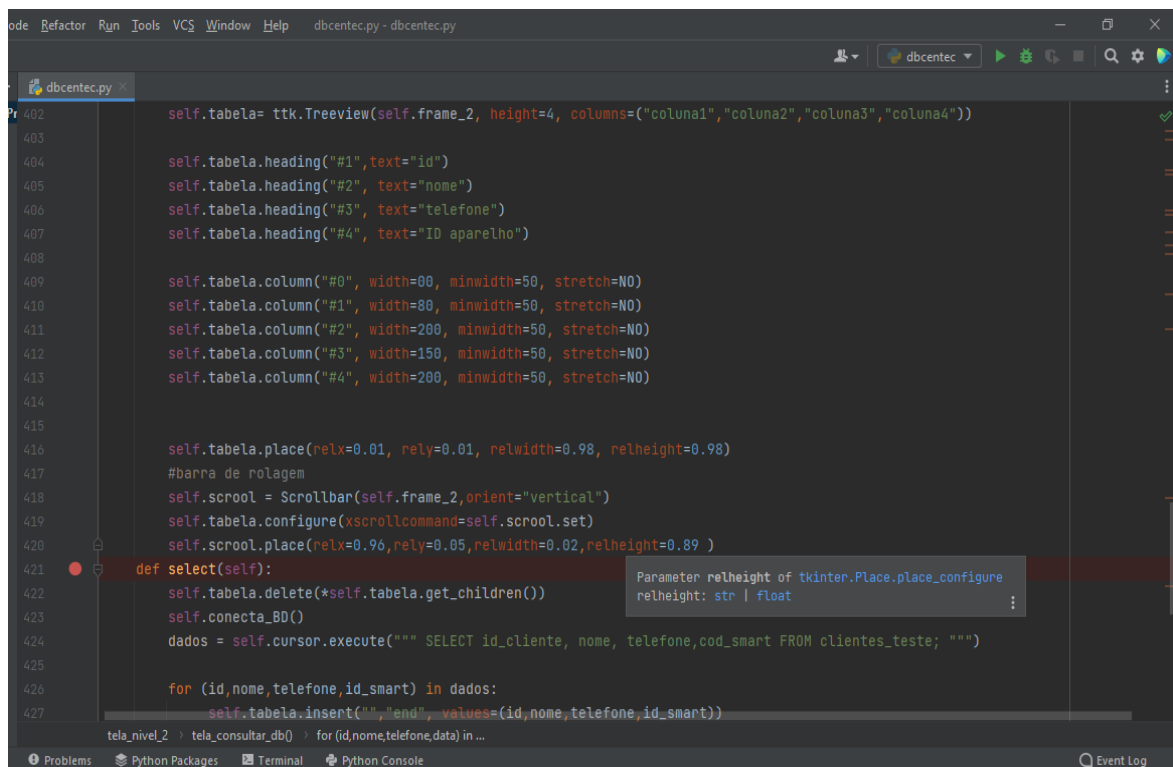


```

375 self.bot_mostra.place(relx=0.05, rely=0.40, relwidth=0.40, relheight=0.20)
376
377 self.bot_mostra = atk.Button3d(self.frame_1, bg="#1EAAAF", text="GARANTIA", command=self.tela_garantia)
378 self.bot_mostra.place(relx=0.05, rely=0.70, relwidth=0.40, relheight=0.20)
379
380 self.bot_aparelhos = atk.Button3d(self.frame_1, bg="#1EAAAF", text="ENTRADA E SAÍDA DE APARELHOS",
381                                 command=self.tela_entrada_saida_aparelhos)
382 self.bot_aparelhos.place(relx=0.50, rely=0.10, relwidth=0.40, relheight=0.20)
383
384 self.bot_aparelhos = atk.Button3d(self.frame_1, bg="#1EAAAF", text="APARELHOS",
385                                 command=self.tela_aparelhos)
386 self.bot_aparelhos.place(relx=0.50, rely=0.40, relwidth=0.40, relheight=0.20)
387
388 self.bot_limpar = atk.Button3d(self.frame_1, bg="#1EAAAF", text="limpar", command= self.limpa_tela_inicial)
389 self.bot_limpar.place(relx=0.70, rely=0.70, relwidth=0.10, relheight=0.20)
390
391 self.bot_buscar = atk.Button3d(self.frame_1, bg="#1EAAAF", text="Buscar", fg="black")
392 self.bot_buscar.place(relx=0.80, rely=0.70, relwidth=0.10, relheight=0.20)
393
394 #criação das label DO NOME
395 self.lb_nome = Label(self.frame_1, text= "Nome \ndo cliente", fg="white", bg="#006675", font=("candara", "10", "bold italic"))
396 self.lb_nome.place(relx=0.50, rely=0.7, relwidth=0.10, relheight=0.20)
397 #codigo de entrada DO NOME
398 self.entrada_nome_i = Entry(self.frame_1)
399 self.entrada_nome_i.place(relx=0.6, rely=0.75, relwidth=0.10, relheight=0.10)
400

```

Figura 4.2.1 < Tela inicial > Autor <cleverson kozuf>

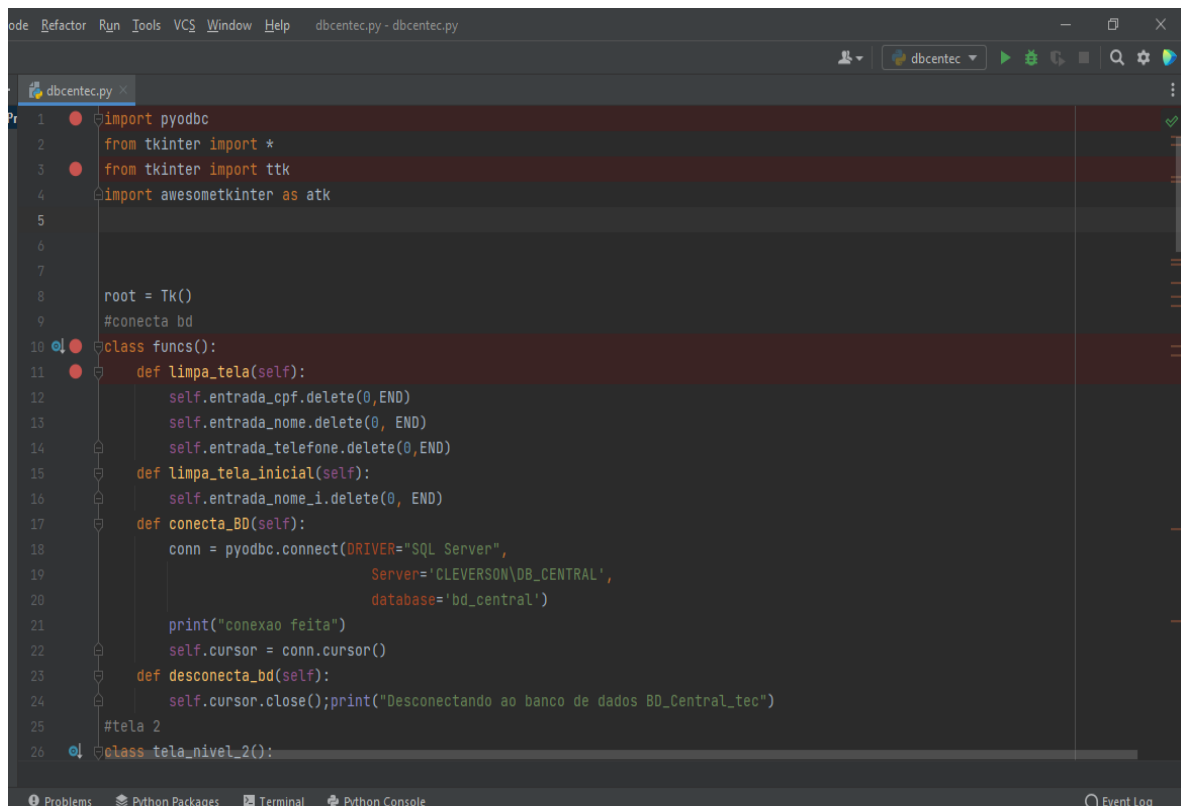


```

402 self.tabela= ttk.Treeview(self.frame_2, height=4, columns=("coluna1", "coluna2", "coluna3", "coluna4"))
403
404 self.tabela.heading("#1", text="id")
405 self.tabela.heading("#2", text="nome")
406 self.tabela.heading("#3", text="telefone")
407 self.tabela.heading("#4", text="ID aparelho")
408
409 self.tabela.column("#0", width=00, minwidth=50, stretch=NO)
410 self.tabela.column("#1", width=80, minwidth=50, stretch=NO)
411 self.tabela.column("#2", width=200, minwidth=50, stretch=NO)
412 self.tabela.column("#3", width=150, minwidth=50, stretch=NO)
413 self.tabela.column("#4", width=200, minwidth=50, stretch=NO)
414
415
416 self.tabela.place(relx=0.01, rely=0.01, relwidth=0.98, relheight=0.98)
417 #barra de rolagem
418 self.scrool = Scrollbar(self.frame_2, orient="vertical")
419 self.tabela.configure(xscrollcommand=self.scrool.set)
420 self.scrool.place(relx=0.96, rely=0.05, relwidth=0.02, relheight=0.89 )
421
422 def select(self):
423     self.tabela.delete(*self.tabela.get_children())
424     self.conecta_BD()
425     dados = self.cursor.execute(""" SELECT id_cliente, nome, telefone, cod_smart FROM clientes_teste; """)
426
427     for (id,nome,telefone,id_smart) in dados:
428         self.tabela.insert("", "end", values=(id,nome,telefone,id_smart))

```

4.2.2 < Tela inicial > Autor <cleverson kozuf>

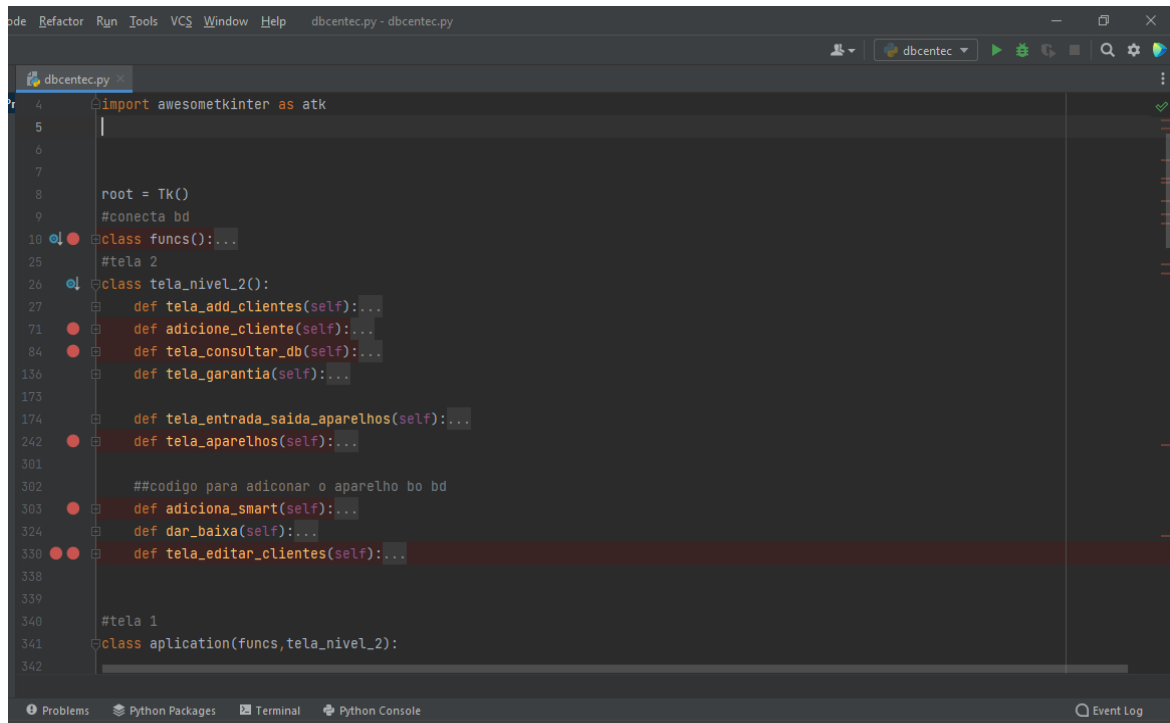


```

1 import pyodbc
2 from tkinter import *
3 from tkinter import ttk
4 import awesometkinter as atk
5
6
7
8 root = Tk()
9 #conecta bd
10 class funcs():
11     def limpa_tela(self):
12         self.entrada_cpf.delete(0,END)
13         self.entrada_nome.delete(0, END)
14         self.entrada_telefone.delete(0,END)
15     def limpa_tela_inicial(self):
16         self.entrada_nome_i.delete(0, END)
17     def conecta_bd(self):
18         conn = pyodbc.connect(DRIVER="SQL Server",
19                               Server='CLEVERSON\DB_CENTRAL',
20                               database='bd_central')
21         print("conexao feita")
22         self.cursor = conn.cursor()
23     def desconecta_bd(self):
24         self.cursor.close();print("Desconectando ao banco de dados BD_Central_tec")
25 #tela 2
26 class tela_nivel_2():

```

Figura 4.3 < def funções > Autor <cleverson kozuf>

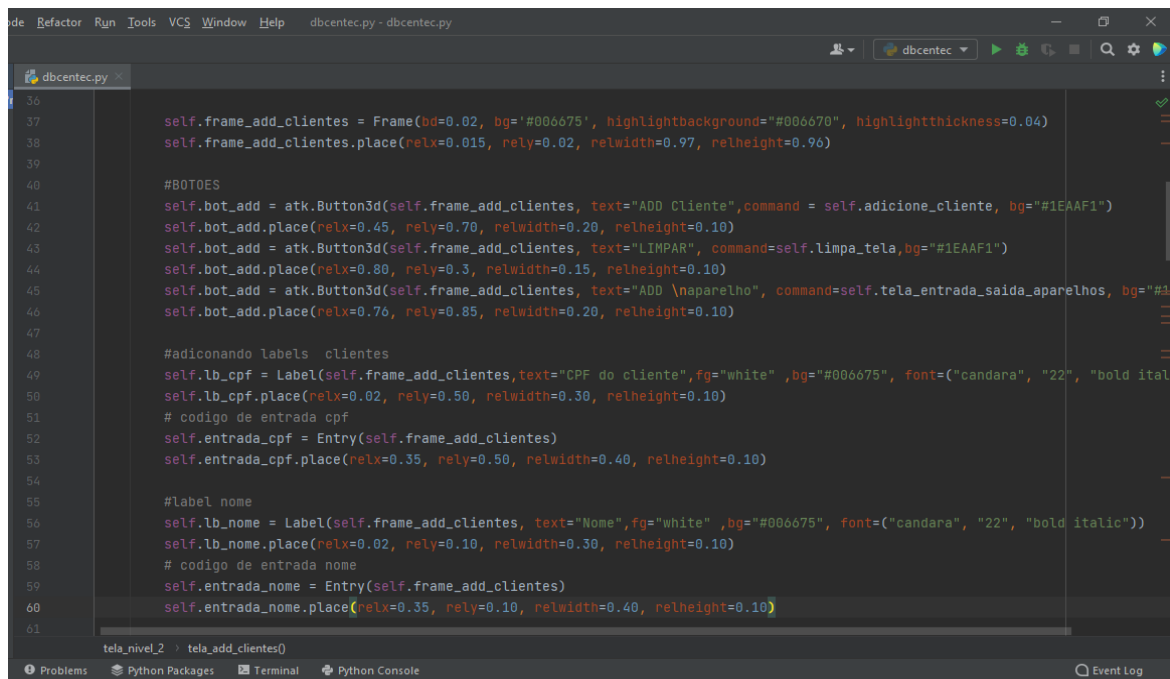


```

4 import awesometkinter as atk
5
6
7
8 root = Tk()
9 #conecta bd
10 class funcs():...
25 #tela 2
26 class tela_nivel_2():
27     def tela_add_clientes(self):...
71     def adicione_cliente(self):...
84     def tela_consultar_db(self):...
136     def tela_garantia(self):...
173
174     def tela_entrada_saida_aparelhos(self):...
242     def tela_aparelhos(self):...
301
302     ##codigo para adicionar o aparelho bo bd
303     def adiciona_smart(self):...
324     def dar_baixa(self):...
330     def tela_editar_clientes(self):...
338
339
340 #tela 1
341 class aplicacion(funcs,tela_nivel_2):
342

```

Figura 4.4 < Telas nível 2 > Autor <cleverson kozuf>

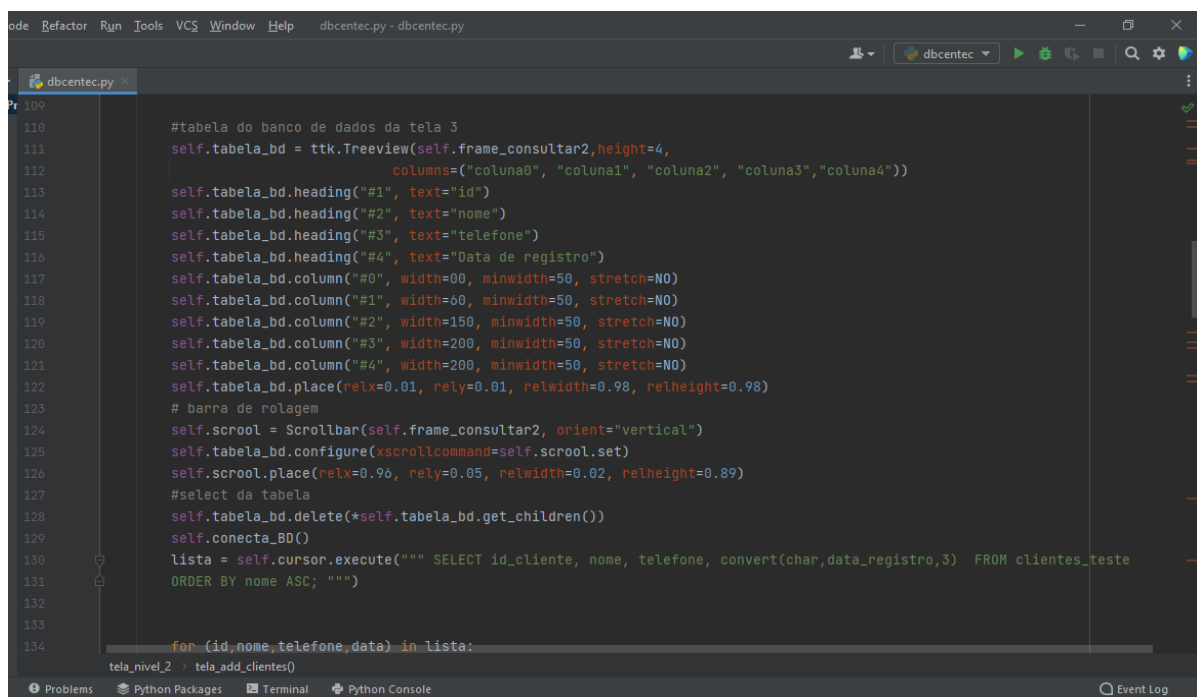


```

36
37     self.frame_add_clientes = Frame(bd=0.02, bg='#006675', highlightbackground="#006670", highlightthickness=0.04)
38     self.frame_add_clientes.place(relx=0.015, rely=0.02, relwidth=0.97, relheight=0.96)
39
40     #BOTOES
41     self.bot_add = atk.Button3d(self.frame_add_clientes, text="ADD Cliente", command = self.adicione_cliente, bg="#1EAAF1")
42     self.bot_add.place(relx=0.45, rely=0.70, relwidth=0.20, relheight=0.10)
43     self.bot_add = atk.Button3d(self.frame_add_clientes, text="LIMPAR", command=self.limpa_tela,bg="#1EAAF1")
44     self.bot_add.place(relx=0.80, rely=0.3, relwidth=0.15, relheight=0.10)
45     self.bot_add = atk.Button3d(self.frame_add_clientes, text="ADD \naparelho", command=self.tela_entrada_saida_aparelhos, bg="#1EAAF1")
46     self.bot_add.place(relx=0.76, rely=0.85, relwidth=0.20, relheight=0.10)
47
48     #adicionando labels clientes
49     self.lb_cpf = Label(self.frame_add_clientes, text="CPF do cliente", fg="white", bg="#006675", font=("candara", "22", "bold italic"))
50     self.lb_cpf.place(relx=0.02, rely=0.50, relwidth=0.30, relheight=0.10)
51     # codigo de entrada cpf
52     self.entrada_cpf = Entry(self.frame_add_clientes)
53     self.entrada_cpf.place(relx=0.35, rely=0.50, relwidth=0.40, relheight=0.10)
54
55     #label nome
56     self.lb_nome = Label(self.frame_add_clientes, text="Nome", fg="white", bg="#006675", font=("candara", "22", "bold italic"))
57     self.lb_nome.place(relx=0.02, rely=0.10, relwidth=0.30, relheight=0.10)
58     # codigo de entrada nome
59     self.entrada_nome = Entry(self.frame_add_clientes)
60     self.entrada_nome.place(relx=0.35, rely=0.10, relwidth=0.40, relheight=0.10)
61
tela_nivel_2 > tela_add_clientes()

```

Figura 4.5 < Tela ADD clientes > Autor <cleverson kozuf>



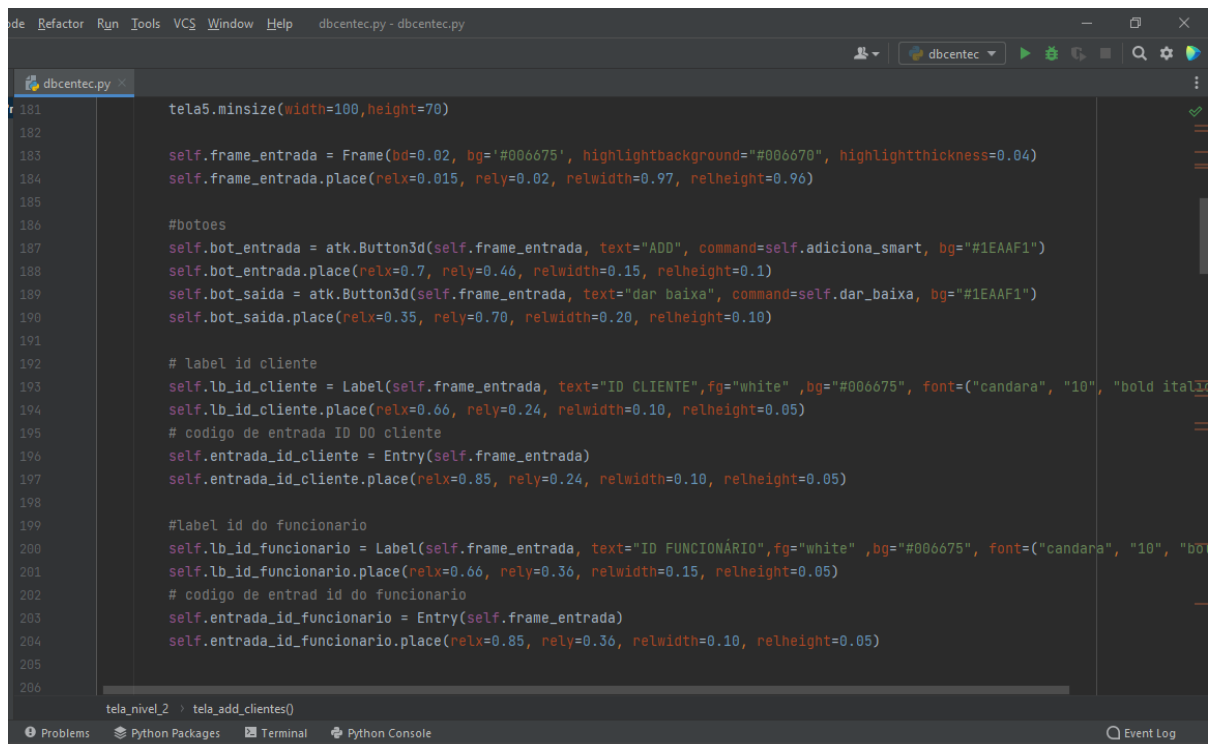
```

109
110     #tabela do banco de dados da tela 3
111     self.tabela_bd = ttk.Treeview(self.frame_consultar2, height=4,
112                                  columns=("coluna0", "coluna1", "coluna2", "coluna3", "coluna4"))
113     self.tabela_bd.heading("#1", text="id")
114     self.tabela_bd.heading("#2", text="nome")
115     self.tabela_bd.heading("#3", text="telefone")
116     self.tabela_bd.heading("#4", text="Data de registro")
117     self.tabela_bd.column("#0", width=00, minwidth=50, stretch=NO)
118     self.tabela_bd.column("#1", width=60, minwidth=50, stretch=NO)
119     self.tabela_bd.column("#2", width=150, minwidth=50, stretch=NO)
120     self.tabela_bd.column("#3", width=200, minwidth=50, stretch=NO)
121     self.tabela_bd.column("#4", width=200, minwidth=50, stretch=NO)
122     self.tabela_bd.place(relx=0.01, rely=0.01, relwidth=0.98, relheight=0.98)
123     # barra de rolagem
124     self.scrtool = Scrollbar(self.frame_consultar2, orient="vertical")
125     self.tabela_bd.configure(xscrollcommand=self.scrtool.set)
126     self.scrtool.place(relx=0.96, rely=0.05, relwidth=0.02, relheight=0.89)
127     #select da tabela
128     self.tabela_bd.delete(*self.tabela_bd.get_children())
129     self.conecta_bd()
130     lista = self.cursor.execute(""" SELECT id_cliente, nome, telefone, convert(char,data_registro,3) FROM clientes_teste
131                                ORDER BY nome ASC; """)
132
133     for (id,nome,telefone,data) in lista:
134
tela_nivel_2 > tela_add_clientes()

```

Figura 4.6 < Tela consultar DB > Autor <cleverson kozuf>



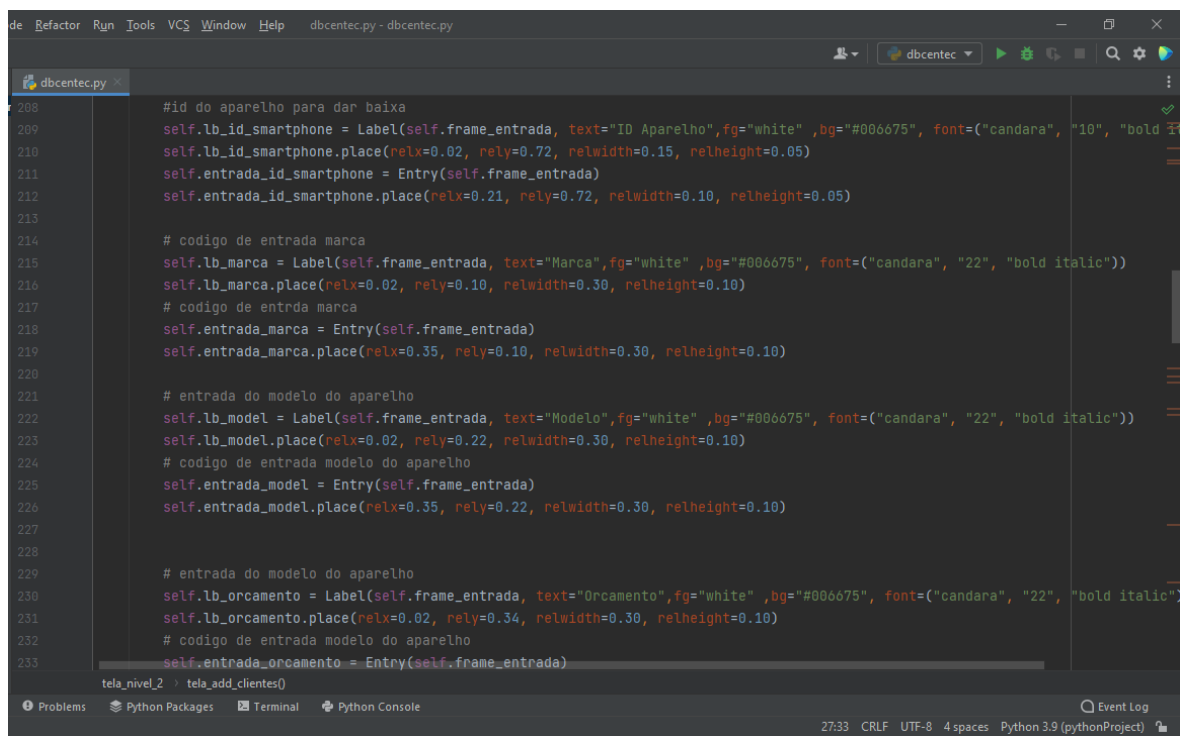


```

181 tela5.minsize(width=100,height=70)
182
183 self.frame_entrada = Frame(bd=0.02, bg="#006675", highlightbackground="#006670", highlightthickness=0.04)
184 self.frame_entrada.place(relx=0.015, rely=0.02, relwidth=0.97, relheight=0.96)
185
186 # botoes
187 self.bot_entrada = atk.Button3d(self.frame_entrada, text="ADD", command=self.adiciona_smart, bg="#1EAAAF")
188 self.bot_entrada.place(relx=0.7, rely=0.46, relwidth=0.15, relheight=0.1)
189 self.bot_saida = atk.Button3d(self.frame_entrada, text="dar baixa", command=self.dar_baixa, bg="#1EAAAF")
190 self.bot_saida.place(relx=0.35, rely=0.70, relwidth=0.20, relheight=0.10)
191
192 # label id cliente
193 self.lb_id_cliente = Label(self.frame_entrada, text="ID CLIENTE", fg="white", bg="#006675", font=("candara", "10", "bold italic"))
194 self.lb_id_cliente.place(relx=0.66, rely=0.24, relwidth=0.10, relheight=0.05)
195 # codigo de entrada ID DO cliente
196 self.entrada_id_cliente = Entry(self.frame_entrada)
197 self.entrada_id_cliente.place(relx=0.85, rely=0.24, relwidth=0.10, relheight=0.05)
198
199 #Label id do funcionario
200 self.lb_id_funcionario = Label(self.frame_entrada, text="ID FUNCIONÁRIO", fg="white", bg="#006675", font=("candara", "10", "bold italic"))
201 self.lb_id_funcionario.place(relx=0.66, rely=0.36, relwidth=0.15, relheight=0.05)
202 # codigo de entrada id do funcionario
203 self.entrada_id_funcionario = Entry(self.frame_entrada)
204 self.entrada_id_funcionario.place(relx=0.85, rely=0.36, relwidth=0.10, relheight=0.05)
205
206

```

Figura 4.7.1< Tela e entrada e saída de aparelhos > Autor <cleverson kozuf>



```

208 #id do aparelho para dar baixa
209 self.lb_id_smartphone = Label(self.frame_entrada, text="ID Aparelho", fg="white", bg="#006675", font=("candara", "10", "bold italic"))
210 self.lb_id_smartphone.place(relx=0.02, rely=0.72, relwidth=0.15, relheight=0.05)
211 self.entrada_id_smartphone = Entry(self.frame_entrada)
212 self.entrada_id_smartphone.place(relx=0.21, rely=0.72, relwidth=0.10, relheight=0.05)
213
214 # codigo de entrada marca
215 self.lb_marca = Label(self.frame_entrada, text="Marca", fg="white", bg="#006675", font=("candara", "22", "bold italic"))
216 self.lb_marca.place(relx=0.02, rely=0.10, relwidth=0.30, relheight=0.10)
217 # codigo de entrada marca
218 self.entrada_marca = Entry(self.frame_entrada)
219 self.entrada_marca.place(relx=0.35, rely=0.10, relwidth=0.30, relheight=0.10)
220
221 # entrada do modelo do aparelho
222 self.lb_model = Label(self.frame_entrada, text="Modelo", fg="white", bg="#006675", font=("candara", "22", "bold italic"))
223 self.lb_model.place(relx=0.02, rely=0.22, relwidth=0.30, relheight=0.10)
224 # codigo de entrada modelo do aparelho
225 self.entrada_model = Entry(self.frame_entrada)
226 self.entrada_model.place(relx=0.35, rely=0.22, relwidth=0.30, relheight=0.10)
227
228 # entrada do modelo do aparelho
229 self.lb_orcamento = Label(self.frame_entrada, text="Orçamento", fg="white", bg="#006675", font=("candara", "22", "bold italic"))
230 self.lb_orcamento.place(relx=0.02, rely=0.34, relwidth=0.30, relheight=0.10)
231 # codigo de entrada modelo do aparelho
232 self.entrada_orcamento = Entry(self.frame_entrada)
233

```

Figura 4.7.2< Tela e entrada e saída de aparelhos > Autor <cleverson kozuf>

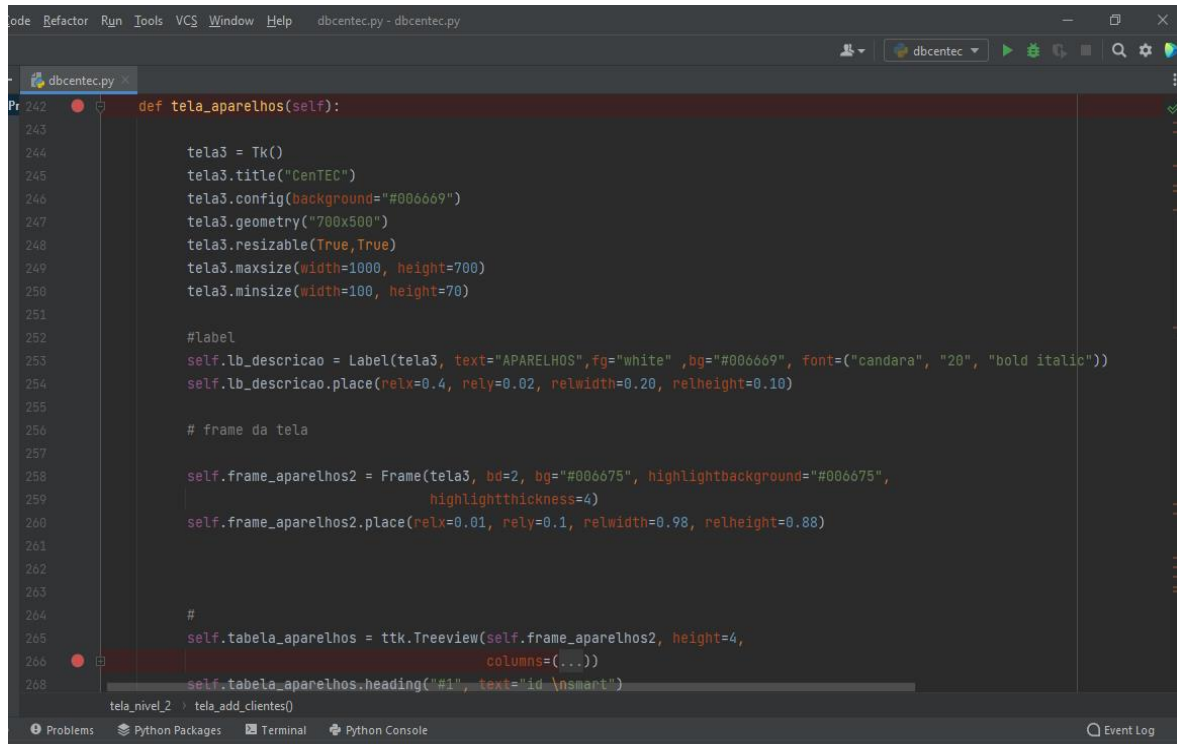


Figura 4.8.1< Tela aparelhos > Autor <cleverson kozuf>

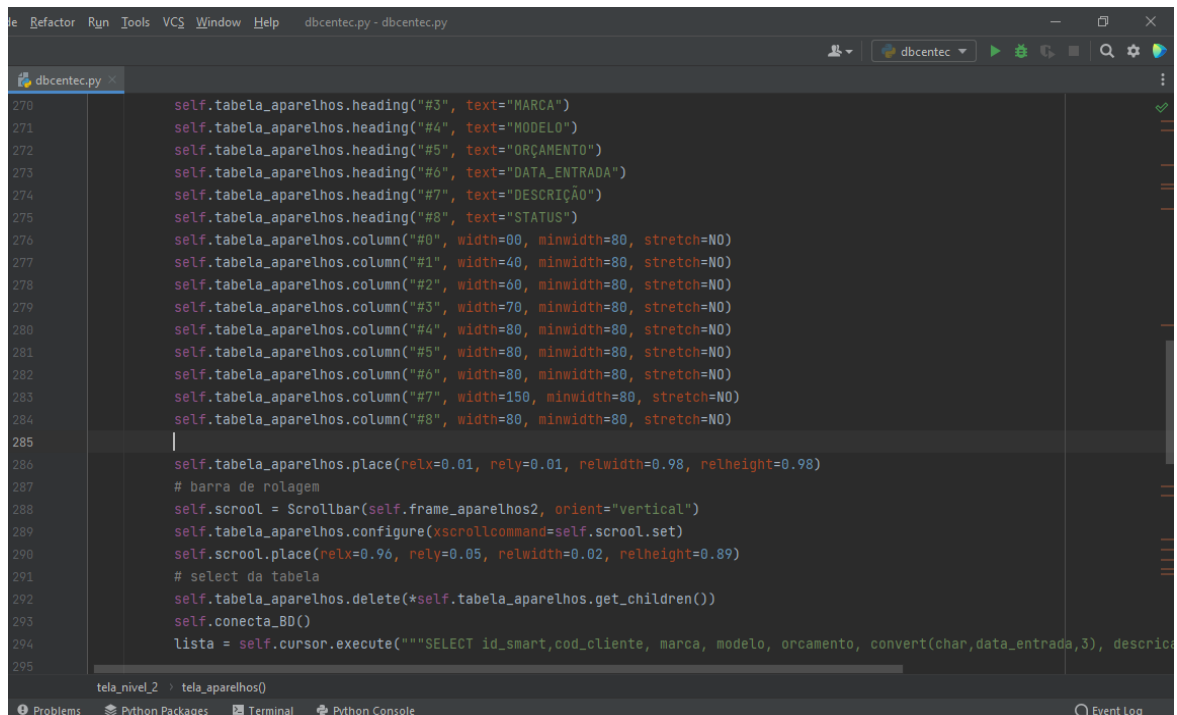


Figura 4.8.2< Tela aparelhos > Autor <cleverson kozuf>

## 4. CONCLUSÃO

Depois de feito todo o estudo do caso da empresa, fazendo o levantamento de informações relevantes e entendendo perfeitamente o funcionamento da empresa e suas regras de negócio, foi possível desenvolver um protótipo simples mas que consegue diminuir muito as dificuldades sofridas pela empresa que foram listadas no início.

Foi feito posteriormente o desenvolvimento do sistema em linguagem python, pois tive mais contato e facilidade de aprender esta linguagem, quando se iniciou o processo de implementação do software foi constatado que houveram várias falhas no desenvolvimento dos diagramas do projeto anterior, muitas delas foram corrigidas, proporcionando o funcionamento pleno do software, como o programa ainda está em desenvolvimento, apresenta alguns “bugs” que serão corrigidos posteriormente, assim como, será feita implementação de várias outras funcionalidades ao decorrer da vida útil do programa.

Os dados e nomes de pessoas apresentadas nas capturas de tela são fictícios, para a proteção da privacidade dos clientes da loja. O programa está em funcionamento para alimentação de seu banco de dados, e posteriormente será feita a correção de eventuais erros que podem ocorrer durante sua execução.

Foi um projeto muito desafiador, consegui compreender a importância de cada etapa de desenvolvimento de um software, tive a oportunidade de praticar linguagens de programação e desenvolver um programa que realmente funciona e que vai auxiliar muito na gestão da loja. Espero ter atingido todos os objetivos propostos pelo projeto integrador.

## REFERENCIAS

AURELIO P, Marcos. Administração de materiais: princípios, conceitos e gestão. 5. ed. Sao paulo: [s. n.], 2010.

DEVMEDIA, Joel (ed.). MER e DER: Modelagem de Bancos de Dados: Modelo Entidade Relacionamento. *In*: Modelo Entidade Relacionamento. [S. l.], 20 maio 2014. Disponível em: <https://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidade-relacionamento-der/14332>. Acesso em: 20 maio 2022.

IDALBERTO, CHIAVENATO. Administração: teoria, processo e prática. 5. ed. Barueri, SP: [s. n.], 2014.  
O QUE é SQL. [S. l.], 14 jul. 2019. Disponível em: <https://www.alura.com.br/artigos/o-que-e-sql>. Acesso em: 26 maio 2022.

H, Ronald. Logística Empresarial: administração de materiais e distribuição física. [S. l.: s. n.], 2015.

VENTURA, Plinio. Exemplos de Requisitos Não Funcionais: O que é um Requisito Não Funcional. *In*: Exemplos de Requisitos Não Funcionais. [S. l.], 31 jan. 2016. Disponível em: <https://www.ateomomento.com.br/exemplos-requisitos-nao-funcionais/>. Acesso em: 9 maio 2022.