RELATÓRIO PROJETO DE ALGORITMO 2023

Nome da empresa: FoneLand.

Equipe: Jonathas Dos Santos Cardoso, Nicolas Serpa Costa, Ryan Rafael, Marcelo Santiago, Caio Antônio Andrade.

16/05/2023

FICR
Faculdade Católica Imaculada Conceição

Sumário

Introdução

O seguinte relatório de projeto, do curso superior de tecnologia em análise e desenvolvimento de sistemas da FICR, tem como objetivo apresentar softwares que ajudem no gerenciamento, e manutenção, do funcionamento interior de uma empresa de smartphones de um shopping center que está em inauguração.

O software trabalhado tem o papel de realizar a gestão do estabelecimento em questão, com o intuito de controlar o nível do estoque, vendas de funcionários, pontos e frequências, ligações com empresas terceirizadas etc.

Algoritmo

1. Login de funcionários

O primeiro algoritmo tem a função de identificar o funcionário através de seu cadastro na loja como gerente, vendedor ou caixa. O login é efetuado com uma senha e nome de usuário especificados para cada tipo de funcionário. Ao logar com sucesso, o usuário é alocado para seu respectivo software de manutenção de tarefas.

O gerente tem seu acesso ao capital da loja, cálculo de lucro mensal, histórico de devolução de valores e trocas de produtos, notas fiscais geradas e gerenciamento do estoque. O vendedor tem acesso ao seu cálculo de comissão e visualização do estoque. E o caixa tem acesso ao próprio software de pagamento e edição da oferta de estoque.

1.1 Sistema de login:

```
def line(valor):
    print('-'*valor)

def login():
    user = input('Usuário: ')
    password = input('Senha: ')

    while validate_user(user, password) == False:
        line(45)
```

```
        print('Usuário ou senha incorretos, tente novamente.')
        line(45)

        user = input('Usuário: ')
        password = input('Senha: ')

        validate_user(user, password)



    employee = validate_user(user, password)

    line(20)
    print(f"{employee} logado")
    line(20)

    return employee

def validate_user(user, password):
    manager = ['gerente','gerente123']
    cashier = ['caixa','caixa123']
    attendant = ['atendente','atendente123']

    if user == manager[0] and password == manager[1]:
        return 'Gerente'

    elif user == cashier[0] and password == cashier[1]:
        return 'Caixista'

    elif user == attendant[0] and password == attendant[1]:
        return 'Atendente'

    else:
        return False

login() # Execução de sistema de login
```

2. Gerenciamento de estoque

O segundo algoritmo tem a função de efetuar a gerência do estoque, registro de recebimento, catalogagem e armazenamento dos produtos. Assim como também de regular, dependendo da presença de nota fiscal e garantia do cliente, e com análise do estado do produto apresentado, o procedimento de troca, devolução, reembolso ou nada sobre o produto apresentado pelo cliente.

2.1 Organização e outras funcionalidades:

```
class LojaSmartphone:
    def _init_(self,nome):
        self.nome = nome
        self.estoque = []
        self.funcionarios = []




    def adicionar_smartphone(self,lista_modelo):
        for modelo in lista_modelo:
            self.estoque.append(modelo)

    def adicionar_funcionario(self,lista_funcionario):
        for funcionario in lista_funcionario:
            self.funcionarios.append(funcionario)

    # funções do estoque ----------------------------------------------------------

    def verificar_estoque_modelos(self):
        for modelo in self.estoque:
            print(f'Modelo: {modelo.nome} | Valor: {modelo.valor} | Unid: {modelo.unid}')

    def valor_total_estoque(self):
        valor_estoque = 0
        for modelo in self.estoque:
            valor_estoque += modelo.valor
        print(f'O valor total do estoque é R${valor_estoque}')

    def achar_modelo_por_nome(self, nome_modelo):
        for modelo in self.estoque:
            if modelo.nome == nome_modelo:
                return modelo

    #  referente a funcionarios ----------------------------------------------------

    def verificar_todos_funcionarios(self):
        for funcionario in self.funcionarios:
                    print(f'Nome: {funcionario.nome} | Cargo: {funcionario.cargo} | Horario do Expediente:
{funcionario.horario}')

    def achar_funcionario_por_nome(self, nome_funcionario):
        for funcionario in self.funcionarios:
            if funcionario.nome == nome_funcionario:
                return funcionario

    #  referente a pagamento -------------------------------------------------------

    def metodo_pagamento(self,nome_modelo):
```

```
print('1. Pix')
print('2. Cartão de Crédito')
print('3. Cartão de Débito')
print('4. Dinheiro Fisico \n')
meio_pag = int(input('digite o meio de pagamento: '))



while meio_pag != 1 and meio_pag != 2 and meio_pag != 3 and meio_pag != 4:
    print('Digite uma opçõa válida!')
    meio_pag = int(input('digite o meio de pagamento:'))

if meio_pag == 1:
    valor_com_desconto = nome_modelo.valor - (nome_modelo.valor * 0.10)

    print('-'*30)
    print(f'Desconto de 10% recebido!')
    print(f'Gerando QR Code no valor de R$ {valor_com_desconto}')
    print('Obrigado pela sua compra, volte sempre!')

if meio_pag == 2:
    print('Pode parcelar em até 12X')
    parcela = int(input('Quantidade de parcelas: '))

    while parcela > 12 or parcela < 0:
        print('Digite um valor  valido!')
        parcela = int(input('Valor da parcelas: '))

    print(f'{parcela} parcelas de R${(nome_modelo.valor / parcela) + (nome_modelo.valor * 0.01):.2f}')
    print('Obrigado pela sua compra, volte sempre!')

if meio_pag == 3:
    valor_com_desconto = nome_modelo.valor - (nome_modelo.valor * 0.10)

    print('-'*30)
    print(f'Desconto de 10% recebido!')
    print(f'Gerando código no valor de R${valor_com_desconto}')
    print('Obrigado pela sua compra, volte sempre!')

if meio_pag == 4:
    valor_com_desconto = nome_modelo.valor - (nome_modelo.valor * 0.10)

    print('-'*30)
    print(f'Desconto de 10% recebido!')
    print(f'Pague o valor de R${valor_com_desconto} no caixa.')
    print('Obrigado pela sua compra, volte sempre!')
```

```
    def efetur_venda(self,nome_funcionario, nome_modelo, unid_vendida):

        funcionario = self.achar_funcionario_por_nome(nome_funcionario)
        modelo = self.achar_modelo_por_nome(nome_modelo)

        self.metodo_pagamento(modelo)




        modelo.unid -= unid_vendida
        funcionario.venda += 1
        funcionario.comissao += modelo.valor * 0.05

        print('-'*50)
        print(f'O funcionario {funcionario.nome} vendeu {unid_vendida} unid(s) de {modelo.nome}')
        print(f'O estoque de {modelo.nome} agora é {modelo.unid}')

    #  referente a bater ponto ----------------------------------------------------------

    def bater_ponto(self,nome_funcionario, horario, horario_entrada):
        funcionario = self.achar_funcionario_por_nome(nome_funcionario)
        funcionario.horario_entrada = horario_entrada
        funcionario.horario = horario

        if funcionario.horario_entrada < funcionario.horario:
            print('Entrou antecipadamente')

        elif funcionario.horario_entrada > funcionario.horario:
            funcionario.desconto_atraso = (funcionario.horario_entrada - funcionario.horario)
            print('Entrou atrasado')
            print(f'O funcionario {funcionario.nome} está com R${funcionario.desconto_atraso:.2f} descontado do
salário')

        else:
            print('Entrou no horario exato.')

        return funcionario

# ------------------------------------------------------------------------------------

class ModeloSmartphone:
    def _init_(self,nome,valor,unid):
        self.nome = nome
        self.valor = valor
        self.unid = unid

class Funcionario:
    def _init_(self, nome, cargo, horario, horario_entrada, venda, comissao, desconto_atraso):
        self.nome = nome
```

```
        self.cargo = cargo
        self.horario = horario
        self.horario_entrada = horario_entrada
        self.venda = venda
        self.comissao = comissao
        self.desconto_atraso = desconto_atraso



loja = LojaSmartphone('FoneLand')

modelos = [
    ModeloSmartphone('Iphone 13',4299.00,120),
    ModeloSmartphone('Iphone 14 pro',7199.00,10),
    ModeloSmartphone('Iphone 12',3699.00,62),
    ModeloSmartphone('Samsung M23',1199.00,43),
    ModeloSmartphone('Samsung S20+',2700.00,3),
    ModeloSmartphone('Samsung A33',2700.00,12),
    ModeloSmartphone('Motorola One Fusion',2700.00,43),
    ModeloSmartphone('Motorola G60',1079.00,30),
    ModeloSmartphone('Motorola G52',1079.00,12),
]

funcionarios = [
    Funcionario('Jonathas','Gerente',0,0,0,0,0),
    Funcionario('Pedro','Caixa',0,0,0,0,0),
    Funcionario('Joao','Caixa',0,0,0,0,0),
    Funcionario('Marcelo','Caixa',0,0,0,0,0),
]

loja.adicionar_smartphone(modelos)
loja.adicionar_funcionario(funcionarios)

loja.efetur_venda('Jonathas','Samsung M23',1) #Exemplo de execução
```

3. Ponto eletrônico de funcionários

O terceiro algoritmo tem função de marcar compromisso com horário de turno por ponto de funcionários da empresa, criando um histórico de presenças na hora e/ou atrasos por funcionário. O algoritmo realiza a avaliação do funcionário em momento de ponto verificando se há algum atraso, se sim, então é marcada sua presença com atraso e boleto ao iniciar horário de chegada do trabalho. Repete-se o mesmo procedimento para marcação do ponto de saída do trabalho, encerrando o expediente do dia.

4. Atendimento do Cliente

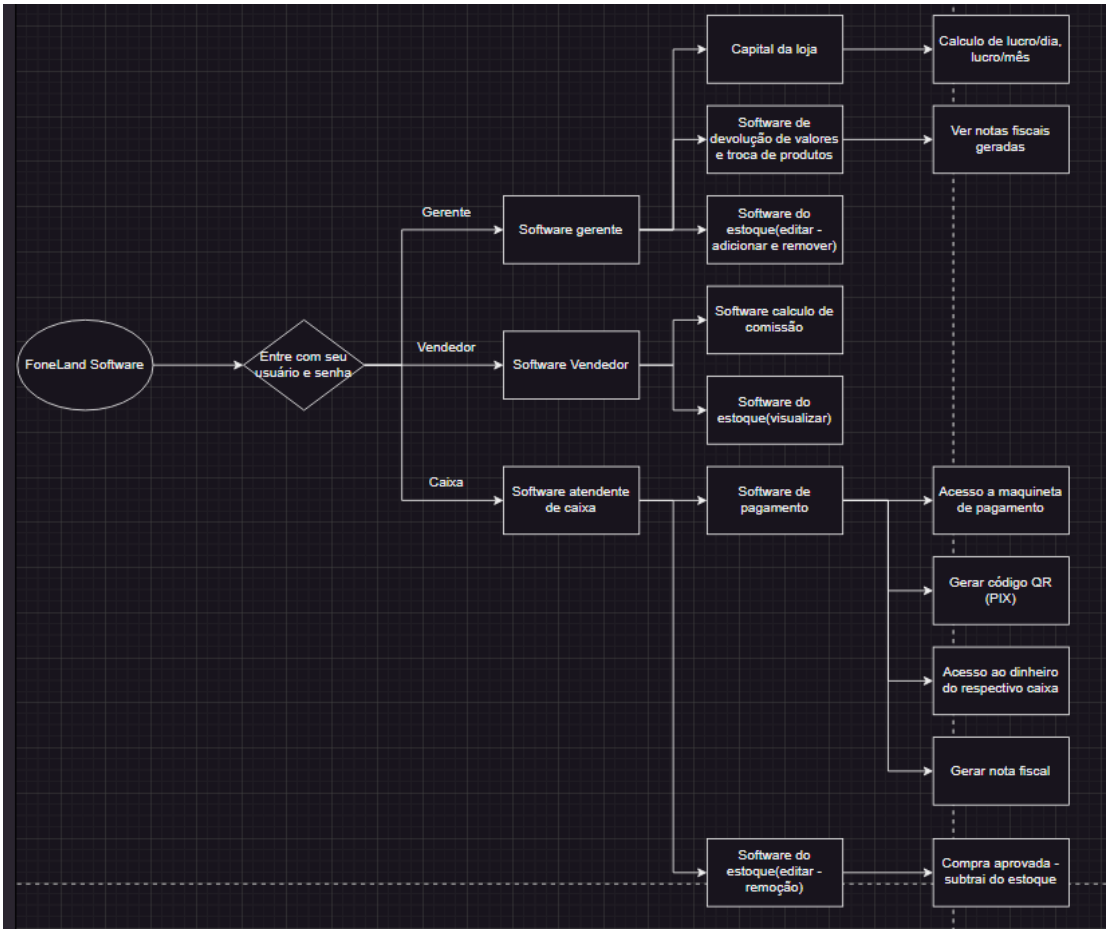
O algoritmo de atendimento esquematiza o procedimento de atender o cliente ao entrar na loja. É realizada a abordagem do cliente pelo vendedor atendente, o vendedor mostra os modelos e ofertas fornecidas e questiona se o cliente deseja comprar algum produto. Caso não queira, é sugerido pelo vendedor outros produtos, e ao fim dos casos por falta de interesse o cliente se retira do estabelecimento.

Caso o cliente se interesse por um produto, então é gerado um pedido de produto e abordado a forma de pagamento (cartão de crédito, cartão de débito, dinheiro ou pix). Caso o valor seja pago à vista, o cliente recebe um desconto do pagamento, caso contrário, é mostrado o valor total com os juros de parcelas. Então é oferecido ao cliente ofertas de seguros e planos, se o mesmo optar por aceitar, é mostrado o valor anterior da compra mais o valor dos seguros e planos, gera-se o código e a validação de pagamento com entrega de produto e nota fiscal.

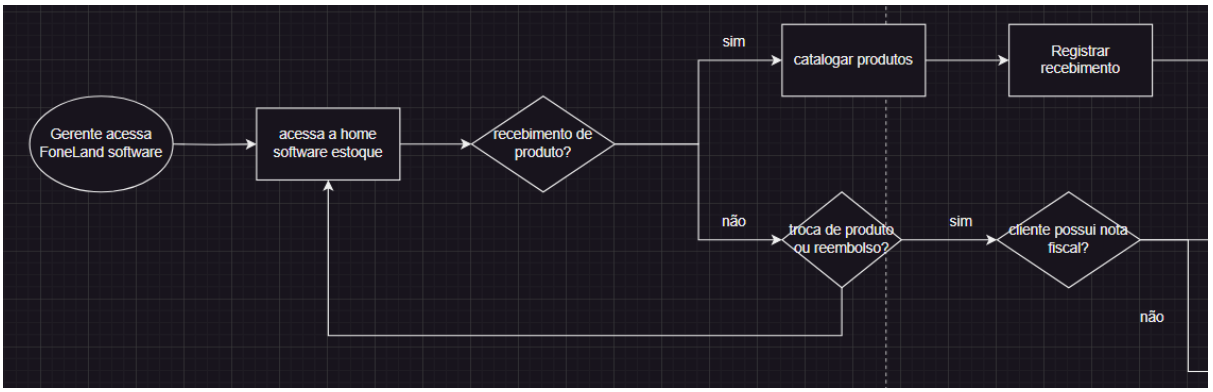
Caso o cliente opte por não aceitar, mantém-se o valor anterior e é realizado o pagamento com código de validação, aprovação e entrega do produto com nota fiscal. Ainda contando que o método de pagamento pode ser alterado para finalização do pedido. Então o cliente se retira do estabelecimento.

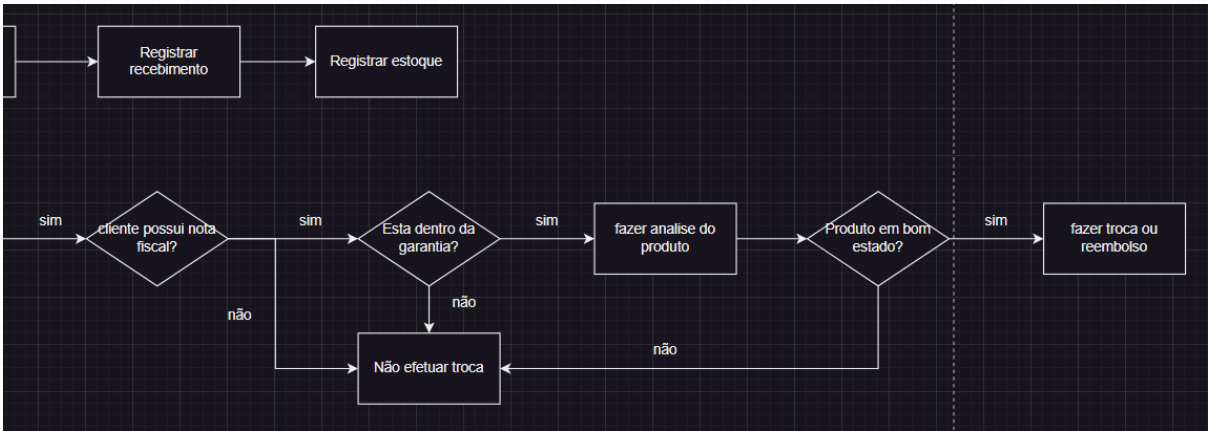
Diagramas

1. Login de funcionários

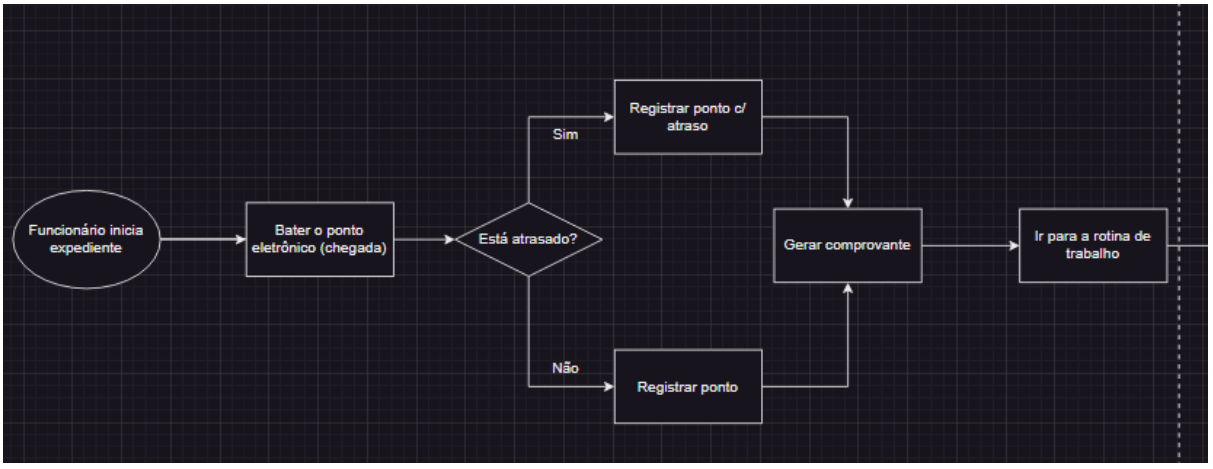


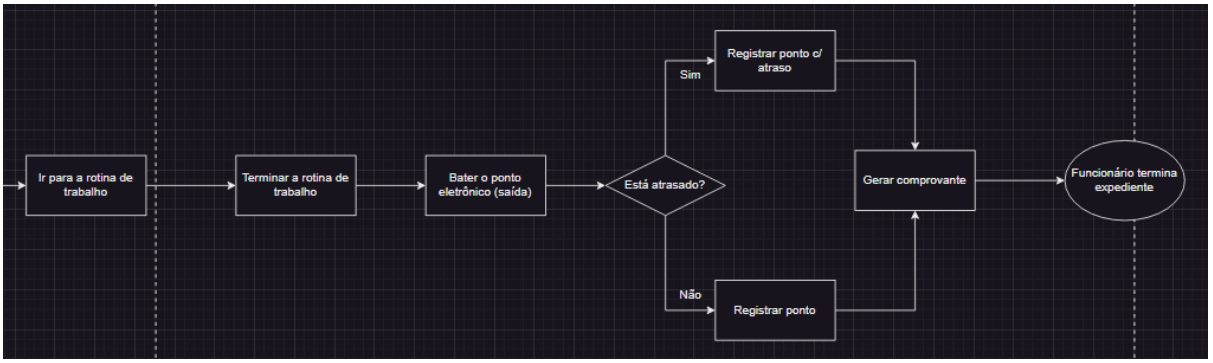
2. Gerenciamento de estoque



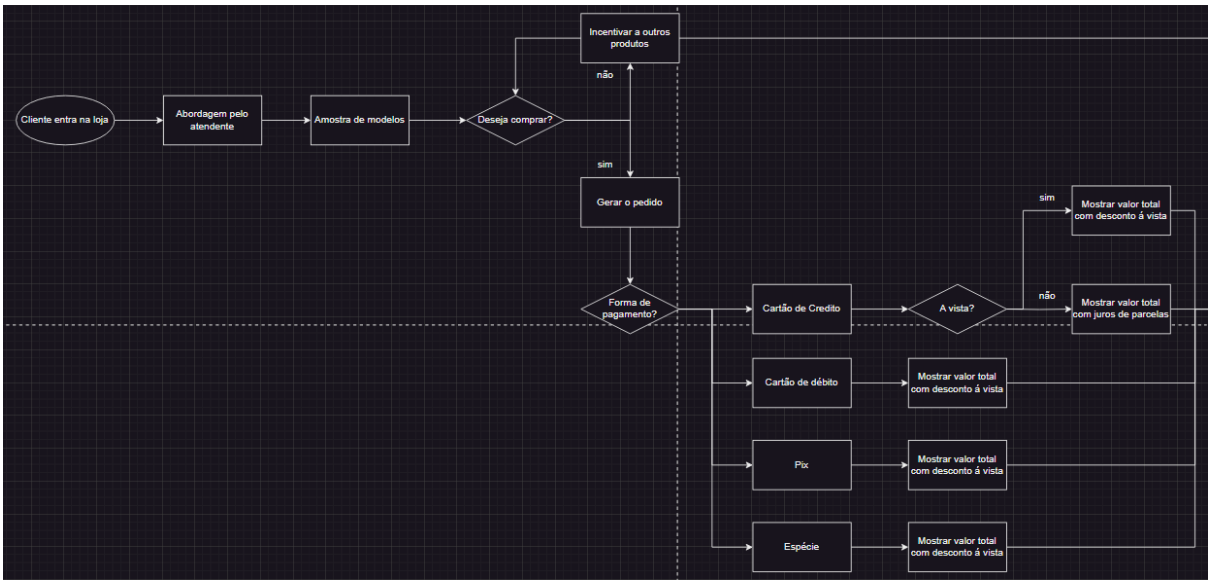


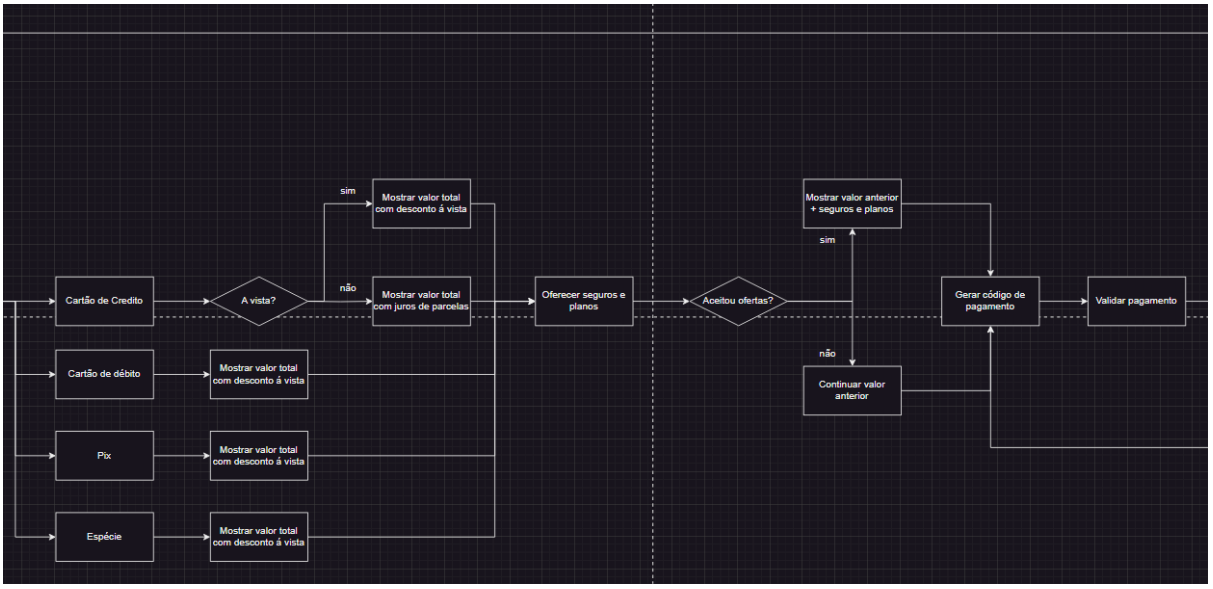
3. Ponto eletrônico de funcionários

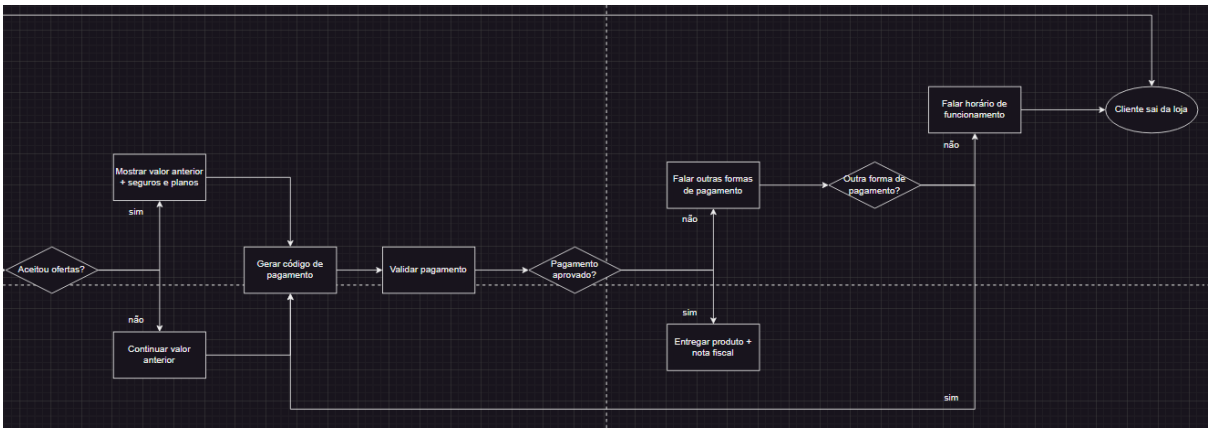




4. Atendimento ao cliente




Cartão de Credito
A vista?
sim
Mostrar valor total com desconto á vista
não
Mostrar valor total com juros de parcelas
Cartão de débito
Mostrar valor total com desconto á vista
Pix
Mostrar valor total com desconto á vista
Espécie
Mostrar valor total com desconto á vista
Oferecer seguros e planos
Aceitou ofertas?
sim
Mostrar valor anterior + seguros e planos
não
Continuar valor anterior
Gerar código de pagamento
Validar pagamento


Aceitou ofertas?
sim
Mostrar valor anterior + seguros e planos
não
Continuar valor anterior
Gerar código de pagamento
Validar pagamento
Pagamento aprovado?
não
Falar outras formas de pagamento
Outra forma de pagamento?
não
Falar horário de funcionamento
Cliente sai da loja
sim
Entregar produto + nota fiscal
sim