



Faculdade de Informática e Administração Paulista

**Giovanna Revito Roz - RM558981**

**Kaian Gustavo de Oliveira Nascimento - RM558986**

**Lucas Kenji Kikuchi - RM554424**

# **Computational Thinking Using Python**

## **Menu Porto**

### **Sprint 1**

# INTEGRANTES

<b>RM</b> <b>(SOMENTE NÚMEROS)</b>	<b>NOME COMPLEMENTO</b> <b>(SEM ABREVIAR)</b>
558981	Giovanna Revito Roz
558986	Kaian Gustavo de Oliveira Nascimento
554424	Lucas Kenji Kikuchi



**SUMÁRIO**

1.Descrição.....	5
2.Funcionalidades.....	7
3. Anexos.....	10

## 1 – Descrição

Nossa plataforma PortoAutoTech visa a implementação de uma interface para auxiliar clientes decorrentes e novos clientes da Porto Seguro, especificamente aqueles que possuem pouco ou nenhum conhecimento sobre problemas relacionados a carros. Nosso objetivo é que, até o fim do projeto, possamos ter uma interface / aplicativo que permita ao cliente ter:

- um autodiagnóstico do problema apresentado pelo veículo através de uma I.A treinada através do Machine Learning em Python, incorporando no Chatbot;
- a possibilidade de se cadastrar com suas informações pessoais (CPF, nome, telefone...) e informações de seu veículo (modelo, marca, ano...) através de nosso aplicativo / website;
- um pré orçamento, com um cálculo baseado no valor do serviço (mão-de-obra + custos adicionais) + valor das peças, e a estimativa de prazo de término do serviço definido com antecedência;
- a localização de oficinas mais próximas, informando a disponibilidade de peças e agendamento de determinados serviços com a integração com um banco de dados da oficina;
- mapeamento do carro, informando através de uma notificação automática quando uma manutenção preventiva deve ser feita, baseado na última vez que o cliente realizou um serviço no carro;
- notificação automática da disponibilidade de uma determinada peça, que será acionada quando a peça for registrada como disponível no banco de dados.
- interface que informa os pontos mais próximos de carregamento para carros elétricos;
- progresso da manutenção, visualizada através de uma barra de progresso, indicando cada mudança importante sobre o status da manutenção, conforme o serviço é feito (manualmente alterado pelo mecânico);
- reconhecimento de imagem através da I.A, que será responsável por analisar a imagem do problema enviado pelo usuário, retornando a solução mais plausível;

- comunicação através de voz (speech-to-text) com o Chatbot, permitindo descrever o problema oralmente ou enviar ruídos emitidos pelo carro, que serão analisados pela I.A;
- ligação em chamada com mecânico através do Chatbot, explicando o progresso da manutenção.

Através das funcionalidades descritas, poderemos auxiliar pessoas que enfrentam dificuldades no entendimento dos problemas do veículo, além de fornecer diferenciais que auxiliarão em necessidades dos clientes que poucas empresas oferecem.

## 2 – Funcionalidades

Até o fim do projeto, Python servirá para o Machine Learning. O algoritmo a ser utilizado é o Random Forest. Através dele, podemos entender o problema do cliente e realizar o autodiagnóstico, fazendo com que a I.A aprenda a lidar com os problemas informados pelo usuário, retornando a saída apropriada para cada caso. Teríamos a integração com um banco de dados contendo as principais causas para um problema que o carro pode apresentar, fornecendo uma solução plausível para o problema do usuário.

Além disso, o auto orçamento será feito com base no autodiagnóstico dado, levando em consideração o valor do serviço (mão-de-obra + custos adicionais) + o valor das peças utilizadas na manutenção. Por fim, teremos a possibilidade de verificar a disponibilidade de determinada peça consultando o banco de dados, permitindo que o cliente consulte se uma peça está disponível ou não no estoque antes de ir à oficina.

Para a Sprint 1, ainda não utilizamos o Python para funcionalidade descrita. Implementamos um menu simples, contendo 8 opções pertinentes para o projeto:

- **1 - Cadastro Usuário:** ao selecionar a opção “Cadastro Usuário”, o usuário terá que digitar as suas informações pessoais: Nome, Idade, CPF, Endereço e Telefone. Caso um dos campos não siga a expressão regular estabelecida, o usuário será induzido a digitar novamente. Caso o usuário já tenha cadastro, ele será redirecionado para o menu principal;
- **2 - Cadastro Veículo:** ao selecionar a opção “Cadastro Veículo”, o usuário terá que preencher as informações de seu veículo (marca, modelo, ano e placa). Caso um dos campos não siga os padrões descritos, o usuário será induzido a preencher novamente. Para as marcas, utilizaremos uma lista reduzida (Chevrolet, Ford, Honda, Volkswagen, Toyota, Hyundai, Fiat, Nissan, Jeep e Renault). O usuário deverá selecionar uma opção de 0 a 9 para a marca de seu carro. Na próxima tela temos 3 exemplos de modelos de carro da marca escolhida (ex: Ká, EcoSport, Focus), em que o usuário terá que escolher uma opção de 0 a 2. O mesmo será feito para o ano do modelo. Caso o usuário já tenha cadastrado o veículo, será redirecionado ao menu principal;

- **3 - Verificar informações de usuário:** ao selecionar a opção “Verificar informações de usuário”, o usuário poderá visualizar seus dados inseridos no Cadastro Usuário. Caso o usuário não tenha cadastro, ele será redirecionado para a opção Cadastro Usuário. Para sair da opção, basta digitar qualquer coisa no input, retornando para o menu principal;
- **4 - Verificar informações do veículo:** ao selecionar a opção “Verificar informações do veículo”, o usuário poderá visualizar as informações de seu veículo inseridos no “Cadastro Veículo”. Caso o usuário não tenha cadastrado, ele será redirecionado para a opção “Cadastro Veículo”. Para sair da opção, basta digitar qualquer coisa no input, retornando para o menu principal;
- **5 - Autodiagnóstico:** ao selecionar a opção de “Autodiagnóstico”, o usuário será induzido a digitar o problema que está enfrentando com o seu veículo. Por meio de if e elif, poderemos, com a ajuda de palavras-chave, descobrir qual o problema que a pessoa está enfrentando, e retornar o motivo de tal dificuldade. Para a primeira Sprint, será mais como um pré-diagnóstico, apenas relatando onde o problema está acontecendo em geral, sem especificar peças a serem substituídas;
- **6 - Auto orçamento:** ao selecionar a opção de “Auto orçamento”, o usuário só poderá receber o orçamento caso já tenha feito o autodiagnóstico (sem o diagnóstico, não sabemos qual o serviço a ser prestado). Ele será redirecionado para o autodiagnóstico. Com o auto orçamento, poderemos descobrir o gasto do usuário, baseado nos custos do serviço;
- **7 - Disponibilidade de peças:** ao selecionar a opção de “Disponibilidade das peças”, o usuário será apresentado a um menu de peças, divididas em suas respectivas categorias. Ao selecionar uma das opções, a aplicação retornará uma lista de 4 peças, referentes a categoria escolhida. Ao lado da peça temos a quantidade disponível. Esses valores serão mantidos em um dicionário, e movidos para um banco de dados em futuras Sprints;
- **0 – Sair:** Por fim, a opção “Sair” fará com que o menu principal feche, finalizando a operação. Qualquer valor fora das opções citadas será dado como inválido, e o usuário terá que digitar uma opção novamente, até que esteja correta.

Para as próximas sprints buscaremos desenvolver novas funcionalidades que incrementem os serviços que nossa aplicação pode oferecer ao usuário, além de incrementar as já existentes.

### 3 – Anexos

#### Menu:

```
1 # menu inicial
2 while True:
3     print("\n===== [ MENU ] =====\n")
4     print("1 - Cadastro Usuário")
5     print("2 - Cadastro Veículo")
6     print("3 - Verificar informações de usuário")
7     print("4 - Verificar informações do veículo")
8     print("5 - Autodiagnóstico")
9     print("6 - Auto orçamento")
10    print("7 - Disponibilidade de peças")
11    print("0 - Sair\n")
12    option = input("Opção: ")
13    if not option.isdigit() or (int(option) > 7 or int(option) < 0):
14        print("Selecione uma opção válida.")
15        continue
16    option = int(option)
17    if option == 0:
18        print("\nSolicitação encerrada.\n")
19        break
20    elif option == 1:
21        cadastro_usuario()
22    elif option == 2:
23        cadastro_veiculo()
24    elif option == 3:
25        verificar_usuario()
26    elif option == 4:
27        verificar_veiculo()
28    elif option == 5:
29        problema = auto_diagnostico()
30    elif option == 6:
31        auto_orcamento()
32    elif option == 7:
33        disponibilidade_pecas()
```

## Função cadastro\_usuario():

```
1 # cadastro do usuario
2 def cadastro_usuario():
3     print("Iniciando cadastro do usuário...\n")
4     if len(usuarioDados) > 0:
5         print("Você já está cadastrado. Irei te redirecionar para o menu...")
6         return
7     while True:
8         nome = input("Digite o seu nome.....: ")
9         if re.match(regexNome, nome) is None:
10            print("Digite um nome válido.")
11            continue
12        else:
13            usuarioDados.append(nome)
14            break
15    while True:
16        idade = input("Digite sua idade.....: ")
17        if not idade.isdigit() or int(idade) < 18:
18            print("Digite uma idade válida.")
19            continue
20        else:
21            usuarioDados.append(idade)
22            break
23    while True:
24        cpf = input("Digite o seu CPF (ex: xxx.xxx.xxx-xx)....: ")
25        if re.match(regexCpf, cpf) is None:
26            print("Digite um CPF válido.")
27            continue
28        else:
29            usuarioDados.append(cpf)
30            break
31    while True:
32        endereco = input("Digite o seu endereço.....: ")
33        if re.match(regexNome, endereco) is None:
34            print("Digite um endereço válido.")
35            continue
36        else:
37            usuarioDados.append(endereco)
38            break
39    while True:
40        telefone = input("Digite o seu telefone (ex: xx xxxxx-xxxx): ")
41        if re.match(regexTel, telefone) is None:
42            print("Digite um número de telefone válido.")
43            continue
44        else:
45            usuarioDados.append(telefone)
46            break
47    print("\nUsuário cadastrado com sucesso!")
```

## Função cadastro\_veiculo():

```
1 #cadastro do veiculo
2 def cadastro_veiculo():
3     print("Iniciando cadastro de veiculo...\n")
4     if len(usuarioVeiculo) > 0:
5         print("Você já possui um veículo cadastrado. Irei te redirecionar para o menu...")
6         return
7     # cadastro marca através de um menu
8     while True:
9         print("=====[ MARCA ]=====\n")
10        for i in range(len(marcas)):
11            print(f"{i:<2} - {list(marcas)[i]}")
12        marca = input("\nSelecione a marca do seu carro.....: ")
13        if not marca.isdigit() or (int(marca) > 10 or int(marca) < 0):
14            print("Selecione uma opção válida.")
15            continue
16        else:
17            marca = int(marca)
18            usuarioVeiculo.append(list(marcas.keys())[marca])
19            break
20    # cadastro modelo através de um menu
21    while True:
22        print("\n=====[ MODELO ]=====\n")
23        for i in range(len(marcas[usuarioVeiculo[0]])):
24            print(f"{i} - {list(marcas[usuarioVeiculo[0]])[i]}")
25        modelo = input("\nSelecione o modelo do seu carro.....: ")
26        if not modelo.isdigit() or (int(modelo) > 2 or int(modelo) < 0):
27            print("Selecione uma opção válida.")
28            continue
29        else:
30            modelo = int(modelo)
31            usuarioVeiculo.append(list(marcas[usuarioVeiculo[0]].keys())[modelo])
32            break
33    # cadastro do ano do veículo
34    while True:
35        print("\n=====[ ANO ]=====\n")
36        for i in range(len(marcas[usuarioVeiculo[0]][usuarioVeiculo[1]])):
37            print(f"{i} - {list(marcas[usuarioVeiculo[0]][usuarioVeiculo[1]])[i]}")
38        ano = input("\nSelecione o ano do seu carro.....: ")
39        if not ano.isdigit() or (int(ano) > 2 or int(ano) < 0):
40            print("Selecione uma opção válida.")
41            continue
42        else:
43            ano = int(ano)
44            usuarioVeiculo.append(marcas[usuarioVeiculo[0]][usuarioVeiculo[1]][ano])
45            break
46    # cadastro da placa do veículo
47    while True:
48        placaVeiculo = input("Qual a placa do seu carro? (ex: ABC-1D23): ")
49        if re.match(regexPlaca, placaVeiculo) is None:
50            print("Insira um formato de placa válido.")
51            continue
52        else:
53            usuarioVeiculo.append(placaVeiculo)
54            break
55    print("\nVeículo foi cadastrado com sucesso!")
```

## Função verificar\_usuario():

```
1 # visualização das informações do usuário
2 def verificar_usuario():
3     print("Iniciando verificação de informações de usuário...")
4     if usuarioDados == []:
5         print("Você ainda não se cadastrou. Irei te redirecionar para a seção de cadastro...\n")
6         cadastro_usuario()
7     else:
8         print("\n===== [ INFORMAÇÕES DO USUÁRIO ] =====\n")
9         print(f"Nome.....: {usuarioDados[0]}")
10        print(f"Idade.....: {usuarioDados[1]}")
11        print(f"CPF.....: {usuarioDados[2]}")
12        print(f"Endereço: {usuarioDados[3]}")
13        print(f"Telefone: {usuarioDados[4]}\n")
14        input("Pressione ENTER para voltar ao menu: ")
15        print("Retornando ao menu principal...")
```

## Função verificar\_veiculo():

```
1 # visualização das informações do veículo
2 def verificar_veiculo():
3     print("Iniciando verificação de informações do veículo...")
4     if usuarioVeiculo == []:
5         print("Você ainda não cadastrou o veículo. Irei te redirecionar para a seção de cadastro...\n")
6         cadastro_veiculo()
7     else:
8         print("\n===== [ INFORMAÇÕES DO VEÍCULO ] =====\n")
9         print(f"Marca.....: {usuarioVeiculo[0].capitalize()}")
10        print(f"Modelo.....: {usuarioVeiculo[1]}")
11        print(f"Ano.....: {usuarioVeiculo[2]}")
12        print(f"Placa.....: {usuarioVeiculo[3]}\n")
13        input("Pressione ENTER para voltar ao menu: ")
14        print("Retornando ao menu principal...")
```

## Função auto\_diagnostico():

```
1 # pré-diagnóstico básico, apenas com if e palavras-chave
2 def auto_diagnostico():
3     print("\nIniciando autodiagnóstico...")
4     while True:
5         problema = (input("\nQual o problema enfrentado no carro?: ")).lower()
6         if 'ar-condicionado' in problema or 'ar condicionado' in problema or 'gelando' in problema or 'esfriando' in problema:
7             print("\nO problema apresentado está relacionado a alguma peça do ar-condicionado.")
8             problema = 'Ar-condicionado'
9             return problema
10        elif 'superaquecimento' in problema or 'motor' in problema:
11            print("\nO problema apresentado está relacionado ao sistema de arrefecimento, possivelmente afetado por uma falha na bomba d'água ou vazamento no radiador.")
12            problema = 'Arrefecimento'
13            return problema
14        elif 'freio' in problema or 'frear' in problema or 'freiar' in problema or 'frenagem' in problema:
15            print("\nO problema apresentado está relacionado aos discos e pastilhas de freio.")
16            problema = 'Freios'
17            return problema
18        elif 'devagar' in problema or 'lento' in problema or 'potência' in problema or 'aceleração' in problema:
19            print("\nO problema apresentado está relacionado ao desempenho do motor, possivelmente afetado por problemas de filtragem ou falhas nas velas de ignição.")
20            problema = 'Filtros e Velas'
21            return problema
22        elif 'volante' in problema or 'tremar' in problema or 'tremendo' in problema or 'vibração' in problema or 'vibrando' in problema or 'pneu' in problema or 'roda' in problema:
23            print("\nO problema apresentado está relacionado ao balanceamento das rodas.")
24            problema = 'Balanceamento'
25            return problema
26        elif 'desalinhado' in problema or 'puxando' in problema or 'volante desalinhado' in problema:
27            print("\nO problema apresentado está relacionado ao desalinhamento das rodas.")
28            problema = 'Alinhamento de Rodas'
29            return problema
30        elif 'não liga' in problema or 'partida' in problema or 'luz fraca' in problema or 'elétrica' in problema:
31            print("\nO problema apresentado está em alguma parte elétrica do carro (bateria, cabos, ignição)")
32            problema = 'Elétrica'
33            return problema
34        elif 'óleo' in problema or 'nível de óleo baixo' in problema or 'vazamento' in problema:
35            print("\nO problema apresentado está relacionado ao óleo do motor do carro.")
36            problema = 'Óleo do Motor'
37            return problema
38        else:
39            print("Problema não registrado.")
40            continue
```

### Função auto\_orcamento():

```
1 # auto-orçamento com base no pré-diagnóstico
2 def auto_orcamento():
3     print("\nIniciando autoorçamento...")
4     while True:
5         # o autoorçamento só será feito caso o problema tenha sido descoberto no pré-diagnóstico.
6         if not problema:
7             print("Você não realizou um autodiagnóstico ainda. Irei te redirecionar para realizar um.")
8             auto_diagnostico()
9             break
10        for servico, preco in servicos_preco.items():
11            if servico == problema:
12                print("\n===== [ ORÇAMENTO ] =====\n")
13                print(f"Problema.....: {servico}")
14                print(f"Preço do serviço...: R${preco:.2f}")
15                input("\nPressione ENTER para voltar ao menu: ")
16                print("Retornando ao menu principal...")
17            break
```

### Função disponibilidade\_pecas():

```
1 # verificar a disponibilidade de determinada peça
2 def disponibilidade_pecas():
3     print("Iniciando verificação da disponibilidade de peça...")
4     while True:
5         print("\n===== [ PEÇAS ] =====\n")
6         for i in range(len(pecas_dados)):
7             print(f"{i+1:<2d} - {list(pecas_dados)[i]}")
8         print("0 - Sair\n")
9         option_pecas = input("Selecione um dos conjuntos para continuar: ")
10        if not option_pecas.isdigit() or (int(option_pecas) > 15 or int(option) < 0):
11            print("\nSelecione uma opção válida.")
12            continue
13        option_pecas = int(option_pecas)
14        if option_pecas == 0:
15            print("Retornando ao menu principal...")
16            break
17        if 1 <= option_pecas <= 15:
18            i = option_pecas - 1
19            # obten a chave da opção selecionada
20            chave = list(pecas_dados.keys())[i]
21            # imprime o conjunto selecionado e as peças, juntamente com a quantidade disponível
22            print(f"\n{chave}\n")
23            for peca, disponivel in pecas_dados[chave].items():
24                if disponivel == 0:
25                    print(f"{peca:<30} - {'Peça indisponível'}")
26                elif disponivel == 1:
27                    print(f"{peca:<30} - {disponivel:>2} disponível")
28                else:
29                    print(f"{peca:<30} - {disponivel:>2} disponíveis")
30            input("\nPressione ENTER para voltar ao menu de peças: ")
31            print("Retornando ao menu de peças...")
```

## Definição de Variáveis e Dicionários:

```
1 # módulo importado para utilização de regex
2 import re
3
4 # declaração de variáveis
5 problema = ''
6 usuarioDados = {}
7 usuarioVeiculo = {}
8
9 # expressões regulares
10 regexCpf = r'^\d(3)\.\d(3)\.\d(3)-\d(2)$'
11 regexId = r'^\d(2) 9\d(4)-\d(4)$'
12 regexNome = r'^[A-Za-zÀ-ÿ'\-\ ]+$'
13 regexPlaca = r'^[A-Z]{3}-\d(1)[A-Z]{1}\d(2)$'
14
15 #peças e a disponibilidade
16 pecas_dados = {
17     'Alinhamento': {'Braco de direção': 10, 'Terminal de direção': 3, 'Coxim do amortecedor': 0, 'Barra axial': 2},
18     'Amortecedor e molas': {'Amortecedor dianteiro': 21, 'Amortecedor traseiro': 10, 'Mola dianteira': 23, 'Mola traseira': 0},
19     'Ar condicionado': {'Compressor': 2, 'Condensador': 4, 'Evaporador': 6, 'Sensores de temperatura': 5},
20     'Arrefecimento': {'Radiador': 10, 'Bomba de água': 20, 'Termostato': 6, 'Mangueira de radiador': 4},
21     'Balanceamento': {'Pneus': 54, 'Rodas': 42, 'Contrapesos': 0, 'Porcas de fixação': 102},
22     'Bateria': {'Bateria 40ah': 10, 'Bateria 60ah': 23, 'Cabos de ligação': 32, 'Cabos da bateria': 54},
23     'Cambagem / caster': {'Cambagem dianteira': 10, 'Cambagem traseira': 13, 'Caster dianteiro': 32, 'Caster traseiro': 14},
24     'Correias do Motor': {'Correia dentada': 2, 'Correia do alternador': 6, 'Correia da direção hidráulica': 0, 'Correia do ar condicionado': 0},
25     'Freio': {'Discos de freio dianteiros': 43, 'Discos de freio traseiros': 53, 'Pastilhas de freio dianteiras': 32, 'Pastilhas de freio traseiras': 54},
26     'Embreagens': {'Disco de embreagem': 5, 'Platô': 4, 'Atuador hidráulico': 10, 'Cilindro mestre': 15},
27     'Filtros': {'Filtro de ar': 30, 'Filtro de óleo': 40, 'Filtro de combustível': 45, 'Filtro de cabine': 23},
28     'Óleo do motor': {'Óleo lubrificante 5W-30': 32, 'Óleo lubrificante 10W-40': 24, 'Filtro de óleo': 12, 'Tampão do cárter': 12},
29     'Palhetas do limpador': {'Palheta do limpador dianteiro': 10, 'Palheta do limpador traseiro': 10, 'Braço do limpador dianteiro': 0, 'Braço do limpador traseiro': 0},
30     'Suspensão': {'Braço oscilante': 10, 'Pivô': 21, 'Barra estabilizadora': 32, 'Batente de suspensão': 10},
31     'Ignição': {'Vela de ignição': 21, 'Bobina de ignição': 32, 'Cabos de vela': 21, 'Sensor de posição': 0}
32 }
33
34 # preços dos serviços
35 servicos_preco = {'Ar-condicionado': 75, 'Arrefecimento': 100, 'Freios': 200, 'Filtros e Velas': 50, 'Balanceamento': 200, 'Alinhamento de Rodas': 75, 'Elétrica': 75}
36
37 #marcas disponiveis
38 marcas = {
39     'Chevrolet': {'Corsa': [1995, 2002, 2009], 'Onix': [2012, 2017, 2021], 'Prisma': [2006, 2013, 2018]},
40     'Ford': {'Ka': [1997, 2005, 2012], 'EcoSport': [2003, 2010, 2018], 'Focus': [1998, 2008, 2015]},
41     'Honda': {'Civic': [1996, 2004, 2011], 'Fit': [2001, 2008, 2015], 'HR-V': [2014, 2017, 2020]},
42     'Volkswagen': {'Gol': [1980, 1998, 2012], 'Polo': [1995, 2002, 2010], 'Jetta': [1984, 1999, 2007]},
43     'Toyota': {'Corolla': [1990, 2002, 2010], 'Hilux': [1998, 2005, 2013], 'Etios': [2010, 2015, 2020]},
44     'Hyundai': {'HB20': [2012, 2016, 2020], 'Tucson': [2004, 2010, 2016], 'Creta': [2014, 2018, 2022]},
45     'Fiat': {'Uno': [1983, 1996, 2004], 'Argo': [2017, 2020, 2023], 'Toro': [2016, 2019, 2022]},
46     'Nissan': {'March': [2002, 2010, 2018], 'Versa': [2006, 2013, 2020], 'Kicks': [2016, 2019, 2023]},
47     'Jeep': {'Renegade': [2014, 2017, 2021], 'Compass': [2006, 2012, 2018], 'Grand Cherokee': [1992, 2005, 2014]},
48     'Renault': {'Kwid': [2015, 2018, 2022], 'Sanderio': [2008, 2014, 2019], 'Duster': [2010, 2015, 2020]}
49 }
```