



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 2

INTEGRANTES

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

SUMÁRIO

1.Descrição.....	5
2.Funcionalidades.....	7
3.Mudanças.....	10
4.Anexos.....	11

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;
- a possibilidade de agendar um serviço através do Chatbot, informando o tipo de serviço, a oficina e a data;
- 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 2, ainda não utilizamos o Python para funcionalidade descrita. Implementamos um menu simples, contendo 11 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 – Gerenciar Usuário:** ao selecionar a opção “Gerenciar Usuário”, o usuário será direcionado a um menu, contendo 3 opções: “Remover Usuário”, que irá remover todos os dados cadastrados; “Visualizar informações do Usuário”, que imprimirá todos os dados (Nome, Idade, CPF, Endereço e Telefone); “Sair”, que retornará o usuário para o menu principal. Essa opção só pode ser acessada se o usuário tiver se cadastrado, caso contrário, será redirecionado ao menu principal.
- **4 - Gerenciar Veículo:** ao selecionar a opção “Gerenciar Veículo”, o usuário será direcionado a um menu, contendo 3 opções: “Remover Veículo”, que irá remover os dados do veículo cadastrado; “Visualizar informações do Veículo”, que imprimirá todos os dados do veículo (Marca, Modelo, Ano e Placa); “Sair”, que retornará o usuário para o menu principal. Essa opção só pode ser acessada se o usuário tiver cadastrado um veículo, caso contrário, será redirecionado ao 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. O usuário só poderá obter o autodiagnóstico caso tenha um veículo cadastrado. 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) e já tenha um veículo cadastrado. 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;

- **8 – Localização das Oficinas:** ao selecionar a opção “Localização das Oficinas”, o usuário será apresentado a uma lista contendo o Bairro/Cidade, e o respectivo endereço da oficina. Ao pressionar uma tecla, retorna ao menu principal;
- **9 – Agendar Serviço:** ao selecionar a opção “Agendar Serviço”, o usuário será direcionado a um menu contendo opções de serviço. Após selecionar o serviço, um menu de oficinas será apresentado, seguido por um menu de datas do agendamento;
- **10 - Gerenciar Serviço:** ao selecionar a opção “Gerenciar Serviço”, o usuário será direcionado a um menu, contendo 3 opções: “Remover agendamento de serviço”, que irá remover os dados do agendamento realizado; “Visualizar informações do agendamento”, que imprimirá todos os dados do agendamento (Serviço, Oficina, Data); “Sair”, que retornará o usuário para o menu principal. Essa opção só pode ser acessada se o usuário tiver realizado um agendamento, caso contrário, será redirecionado ao menu principal.
- **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 – Mudanças

Para a Sprint 2, modificamos algumas funcionalidades do projeto, buscando torná-lo mais funcional e prático ao usuário, corrigindo problemas de validação e adicionando funcionalidades novas. Podemos citar as principais mudanças sendo:

- Passagem de parâmetros nas funções `gerenciar_usuario()`, `gerenciar_veiculo()`, `auto_diagnostico()`, `auto_orcamento()`, `agendar_servico()` e `gerenciar_servico()`;
- Na opção de autodiagnóstico, impedir o usuário de realizá-lo caso não tenha um veículo e usuário cadastrados;
- Na opção de auto orçamento, impedir o usuário de realizá-lo caso não tenha veículo e usuário cadastrados e um autodiagnóstico realizado;
- Correção da validação da idade (maior que 90 é inválido);
- Correção na opção de marca de veículo (maior que 9 é inválido);
- Remoção das funções `verificar_usuario()` e `verificar_veiculo()`;
- Criação das funções `deletar_usuario()` e `deletar_veiculo()`, responsáveis por deletar os dados de usuário e veículo;
- Criação das funções `gerenciar_usuario()` e `gerenciar_veiculo()`, responsáveis pela remoção e verificação dos dados de usuário e veículo;
- Criação da função `localizacao_oficinas()`, responsável por imprimir ao usuário a localização das oficinas da Porto;
- Criação da função `deletar_servico()`, responsável por deletar um serviço agendado;
- Criação da função `gerenciar_servico()`, responsável pela remoção e visualização dos dados do agendamento;
- Criação da lista `dadosServico`, para armazenamento das informações do agendamento feito pelo usuário.

4 – 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 - Gerenciar Usuário")
7     print("4 - Gerenciar Veículo")
8     print("5 - Autodiagnóstico")
9     print("6 - Auto orçamento")
10    print("7 - Disponibilidade de peças")
11    print("8 - Localização das oficinas")
12    print("9 - Agendar Serviço")
13    print("10 - Gerenciar Serviço")
14    print("0 - Sair\n")
15    option = input("Opção: ")
16    if not option.isdigit() or (int(option) > 10 or int(option) < 0):
17        print("\nSelecione uma opção válida.")
18        continue
19    option = int(option)
20    if option == 0:
21        print("\nSolicitação encerrada.\n")
22        break
23    elif option == 1:
24        cadastro_usuario()
25    elif option == 2:
26        cadastro_veiculo()
27    elif option == 3:
28        gerenciar_usuario(usuarioDados)
29    elif option == 4:
30        gerenciar_veiculo(usuarioVeiculo)
31    elif option == 5:
32        problema = auto_diagnostico(usuarioVeiculo, usuarioDados)
33    elif option == 6:
34        auto_orcamento(problema, usuarioVeiculo, usuarioDados)
35    elif option == 7:
36        disponibilidade_peças()
37    elif option == 8:
38        localizacao_oficinas()
39    elif option == 9:
40        agendar_servico(usuarioDados, usuarioVeiculo)
41    elif option == 10:
42        gerenciar_servico(dadosServico)
```

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      # cadastro nome
8      while True:
9          nome = input("Digite o seu nome.....: ")
10         if re.match(regexNome, nome) is None:
11             print("Digite um nome válido.")
12             continue
13         else:
14             usuarioDados.append(nome)
15             break
16     # cadastro idade
17     while True:
18         idade = input("Digite sua idade.....: ")
19         if not idade.isdigit() or int(idade) < 18 or int(idade) > 90:
20             print("Digite uma idade válida.")
21             continue
22         else:
23             usuarioDados.append(idade)
24             break
25     # cadastro CPF
26     while True:
27         cpf = input("Digite o seu CPF (ex: xxx.xxx.xxx-xx)....: ")
28         if re.match(regexCpf, cpf) is None:
29             print("Digite um CPF válido.")
30             continue
31         else:
32             usuarioDados.append(cpf)
33             break
34     # cadastro endereço
35     while True:
36         endereco = input("Digite o seu endereço.....: ")
37         if re.match(regexNome, endereco) is None:
38             print("Digite um endereço válido.")
39             continue
40         else:
41             usuarioDados.append(endereco)
42             break
43     # cadastro telefone
44     while True:
45         telefone = input("Digite o seu telefone (ex: xx xxxxx-xxxx): ")
46         if re.match(regexTel, telefone) is None:
47             print("Digite um número de telefone válido.")
48             continue
49         else:
50             usuarioDados.append(telefone)
51             break
52     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 veiculo 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+1} - {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("\nSelecione uma opção válida.\n")
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+1} - {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("\nSelecione 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 veiculo
34     while True:
35         print("\n===== [ ANO ] =====\n")
36         for i in range(len(marcas[usuarioVeiculo[0]][usuarioVeiculo[1]])):
37             print(f"{i+1} - {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("\nSelecione 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 veiculo
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("\nVeiculo foi cadastrado com sucesso!")
```

Função deletar_usuario() e deletar_veiculo():

```
1 #remover usuario
2 def deletar_usuario():
3     while True:
4         if not usuarioDados:
5             print("\nVocê não está cadastrado ainda.")
6             break
7         op_delete = input("\nDeseja realmente deletar o seu cadastro? S ou N: ")
8         if op_delete.upper() != "S" and op_delete.upper() != "N":
9             print("\nDigite uma opção válida.")
10            continue
11        elif op_delete.upper() == "S":
12            usuarioDados.clear()
13            print("\nCadastro removido com sucesso.")
14            break
15        elif op_delete.upper() == "N":
16            print("\nO cadastro não foi removido.")
17            break
```

```
1 #remover veiculo
2 def deletar_veiculo():
3     while True:
4         if not usuarioVeiculo:
5             print("\nVocê não possui nenhum veículo registrado.")
6             break
7         op_delete = input("\nDeseja realmente remover o veículo? S ou N: ")
8         if op_delete.upper() != "S" and op_delete.upper() != "N":
9             print("\nDigite uma opção válida.")
10            continue
11        elif op_delete.upper() == "S":
12            usuarioVeiculo.clear()
13            print("\nVeículo removido com sucesso.")
14            break
15        elif op_delete.upper() == "N":
16            print("\nO veículo não foi removido.")
17            break
```

Função gerenciar_usuario():

```
1 # visualização das informações do usuário (Nome, Idade, CPF, endereço e telefone)
2 def gerenciar_usuario(usuarioDados):
3     print("\nIniciando menu de gerenciamento do usuário...")
4     while True:
5         if usuarioDados == []:
6             print("\nVocê ainda não se cadastrou.")
7             break
8         print("\n===== [ GERENCIAMENTO USUÁRIO ] =====\n")
9         print("1 - Remover usuário")
10        print("2 - Visualizar informações do usuário")
11        print("0 - Sair")
12        verif_usuario_op = input("\nSelecione uma opção: ")
13        if not verif_usuario_op.isdigit() or int(verif_usuario_op) > 2 or int(verif_usuario_op) < 0:
14            print("\nSelecione uma opção válida.")
15            continue
16        verif_usuario_op = int(verif_usuario_op)
17        if verif_usuario_op == 0:
18            break
19        elif verif_usuario_op == 1:
20            deletar_usuario()
21            break
22        elif verif_usuario_op == 2:
23            print("\n===== [ INFORMAÇÕES DO USUÁRIO ] =====\n")
24            print(f"Nome..... {usuarioDados[0]}")
25            print(f"Idade..... {usuarioDados[1]}")
26            print(f"CPF..... {usuarioDados[2]}")
27            print(f"Endereço: {usuarioDados[3]}")
28            print(f"Telefone: {usuarioDados[4]}\n")
29            input("Pressione ENTER para voltar ao menu: ")
30            print("Retornando ao menu do usuário...")
31            continue
```

Função gerenciar_veiculo():

```
1  # visualização das informações do veículo (Marca, modelo, ano e placa)
2  def gerenciar_veiculo(usuarioVeiculo):
3      print("\nIniciando menu de gerenciamento do veículo...")
4      while True:
5          if usuarioVeiculo == []:
6              print("\nVocê ainda não cadastrou o veículo.")
7              break
8          print("\n===== [ GERENCIAMENTO VEÍCULO ] =====\n")
9          print("1 - Remover veículo")
10         print("2 - Visualizar informações do veículo")
11         print("0 - Sair")
12         verif_veic_op = input("\nSelecione uma opção: ")
13         if not verif_veic_op.isdigit() or int(verif_veic_op) > 2 or int(verif_veic_op) < 0:
14             print("\nSelecione uma opção válida.")
15             continue
16         verif_veic_op = int(verif_veic_op)
17         if verif_veic_op == 0:
18             break
19         elif verif_veic_op == 1:
20             deletar_veiculo()
21             break
22         elif verif_veic_op == 2:
23             print("\n===== [ INFORMAÇÕES DO VEÍCULO ] =====\n")
24             print(f"Marca.....: {usuarioVeiculo[0].capitalize()}")
25             print(f"Modelo.....: {usuarioVeiculo[1]}")
26             print(f"Ano.....: {usuarioVeiculo[2]}")
27             print(f"Placa.....: {usuarioVeiculo[3]}\n")
28             input("Pressione ENTER para voltar ao menu: ")
29             print("Retornando ao menu do veículo...")
30             continue
```


Função auto_diagnostico():


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

Função auto_orcamento():

```
1  # auto-orçamento com base no pré-diagnóstico
2  def auto_orcamento(problemaUsuario, usuarioVeiculo, usuarioDados):
3      while True:
4          print("\nIniciando auto orçamento...")
5          # o autoorçamento só será feito caso o problema tenha sido descoberto no pré-diagnóstico.
6          if usuarioVeiculo == [] and usuarioDados == []:
7              print("\nVocê não possui um veículo cadastrado e nem se cadastrou.")
8              break
9          elif usuarioVeiculo == []:
10             print("\nVocê não cadastrou nenhum veículo.")
11             break
12          elif usuarioDados == []:
13              print("\nVocê não se cadastrou.")
14              break
15          elif not problemaUsuario:
16              print("\nVocê não fez um autodiagnóstico ainda.")
17              break
18          else:
19              for servico, preco in servicos_preco.items():
20                  if servico == problemaUsuario:
21                      print("\n===== [ ORÇAMENTO ] =====\n")
22                      print(f"Problema.....: {servico}")
23                      print(f"Preço do serviço...: R${preco:.2f}")
24                      input("\nPressione ENTER para voltar ao menu: ")
25                      print("Retornando ao menu principal...")
26              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            # obtem 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...")
```

Função localizacao_oficinas():

```
1 def localizacao_oficinas():
2     while True:
3         print("\n===== [ OFICINAS ] =====\n")
4         for oficina, localizacao in oficinas.items():
5             print(f"{oficina:<20}: {localizacao}")
6         input("\nPressione ENTER para voltar ao menu principal: ")
7         print("Retornando ao menu principal...")
8         break
```

Função `agendar_servico()`:

```
1 # agenda um serviço para o usuário
2 def agendar_servico(usuarioDados, usuarioVeiculo):
3     print("Iniciando agendamento do serviço...")
4     while True:
5         if usuarioDados == [] and usuarioVeiculo == []:
6             print("\nVocê ainda não se cadastrou nem cadastrou um veículo.")
7             break
8         elif usuarioVeiculo == []:
9             print("\nVocê ainda não cadastrou um veículo.")
10            break
11        elif usuarioDados == []:
12            print("\nVocê ainda não se cadastrou.")
13            break
14        elif dadosServico != []:
15            print("\nVocê já possui um agendamento marcado.")
16            break
17        while True:
18            print("\n===== [ SERVIÇOS ] =====\n")
19            for i in range(len(list(servicos_preco))):
20                print(f"{i} - {list(servicos_preco)[i]}")
21            op_servico = input("\nQual serviço deseja realizar?: ")
22            if not op_servico.isdigit() or int(op_servico) > 7 or int(op_servico) < 0:
23                print("\nSelecione uma opção válida.")
24                continue
25            else:
26                op_servico = int(op_servico)
27                dadosServico.append(list(servicos_preco)[op_servico])
28                break
29        while True:
30            print("\n===== [ OFICINAS ] =====\n")
31            for i in range(len(list(oficinas))):
32                print(f"{i} - {list(oficinas)[i]}")
33            op_oficina = input("\nEm qual oficina deseja realizar o serviço?: ")
34            if not op_oficina.isdigit() or int(op_oficina) > 9 or int(op_oficina) < 0:
35                print("\nSelecione uma opção válida.")
36                continue
37            else:
38                op_oficina = int(op_oficina)
39                dadosServico.append(list(oficinas)[op_oficina])
40                break
41        while True:
42            print("\n===== [ DATAS ] =====\n")
43            for i in range(len(list(datas))):
44                print(f"{i} - {list(datas)[i]}")
45            op_data = input("\nQual a data que deseja realizar o serviço?: ")
46            if not op_data.isdigit() or int(op_data) > 6 or int(op_data) < 0:
47                print("\nSelecione uma opção válida.")
48                continue
49            else:
50                op_data = int(op_data)
51                dadosServico.append(datas[op_data])
52                break
53    print("\nServiço agendado com sucesso!")
54    break
```

Função deletar_agendamento():

```
1 # deletar o agendamento feito
2 def deletar_agendamento():
3     while True:
4         if not dadosServico:
5             print("\nVocê não agendou um serviço ainda.")
6             break
7         op_delete = input("\nDeseja realmente remover o seu agendamento? S ou N: ")
8         if op_delete.upper() != "S" and op_delete.upper() != "N":
9             print("\nDigite uma opção válida.")
10            continue
11        elif op_delete.upper() == "S":
12            dadosServico.clear()
13            print("\nAgendamento removido com sucesso.")
14            break
15        elif op_delete.upper() == "N":
16            print("\nO agendamento não foi removido.")
17            break
```

Função gerenciar_servico():

```
1 # gerenciar o serviço agendado
2 def gerenciar_servico(dadosServico):
3     print("\nIniciando menu de gerenciamento do serviço...")
4     while True:
5         if dadosServico == []:
6             print("\nVocê ainda não agendou um serviço.")
7             break
8         print("\n===== [ GERENCIAMENTO SERVIÇO ] =====\n")
9         print("1 - Remover agendamento de serviço")
10        print("2 - Visualizar informações do agendamento")
11        print("0 - Sair")
12        verif_gerenc_op = input("\nSelecione uma opção: ")
13        if not verif_gerenc_op.isdigit() or int(verif_gerenc_op) > 2 or int(verif_gerenc_op) < 0:
14            print("\nSelecione uma opção válida.")
15            continue
16        verif_gerenc_op = int(verif_gerenc_op)
17        if verif_gerenc_op == 0:
18            break
19        elif verif_gerenc_op == 1:
20            deletar_agendamento()
21            break
22        elif verif_gerenc_op == 2:
23            print("\n===== [ INFORMAÇÕES DO AGENDAMENTO ] =====\n")
24            print(f"Serviço.....: {dadosServico[0]}")
25            print(f"Oficina.....: {dadosServico[1]}")
26            print(f>Data.....: {dadosServico[2]}")
27            input("\nPressione ENTER para voltar ao menu: ")
28            print("\nRetornando ao menu do serviço...")
29            continue
```

Definição de Variáveis, Dicionários e Listas:

```
1 # módulo importado para utilização de regex
2 import re
3
4 # declaração de variáveis
5 problema = ''
6 usuarioDados = []
7 usuarioVeiculo = []
8 dadosServico = []
9 # expressões regulares
10 regexCpf = r'^\d(3)\.\d(3)\.\d(3)-\d(2)$'
11 regexTel = r'^\d(2) 9\d(4)-\d(4)$'
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': {'Braço 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': 22, '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, 'Óleo do Motor': 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, 2019], '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], 'Sanderó': [2008, 2014, 2019], 'Duster': [2010, 2015, 2020]}
49 }
50
51 # localização de oficinas da Porto
52 oficinas = {
53     'Lins de Vasconcelos': 'Av. Lins de Vasconcelos, 2474 - Vila Mariana, São Paulo - SP, 04112-001',
54     'Conceição': 'Av. Diederichsen, 1426 - Vila Guarani (Zona Sul), São Paulo - SP, 04310-001',
55     'São Caetano do Sul': 'Av. Sen. Roberto Simonsen, 1305 - Cerâmica, São Caetano do Sul - SP, 09530-402',
56     'Jardim Paulista': 'Av. Brigadeiro Luís Antônio, 3383 - Jardim Paulista, São Paulo - SP, 01401-001',
57     'Ipiranga': 'R. Silva Bueno, 1176 - Ipiranga, São Paulo - SP, 04208-000',
58     'Bosque da Saúde': 'Av. Bosque da Saúde, 1276 - Jabaquara, São Paulo - SP, 04142-082',
59     'Vila Mariana': 'R. França Pinto, 1115 - Vila Mariana, São Paulo - SP, 04016-034',
60     'Campo Belo': 'R. Otávio Tarquínio de Sousa, 304 - Campo Belo, São Paulo - SP, 04613-000',
61     'Planalto Paulista': 'Av. dos Bandeirantes, 5024 - Planalto Paulista, São Paulo - SP, 04071-000',
62     'Mooça': 'Rua dos Trilhos, 1327 - Alto da Mooça, São Paulo - SP, 03168-009'
63 }
64
65 # datas ficticias para servicos
66 datas = ['20/05', '21/05', '22/05', '23/05', '24/05', '25/05', '26/05']
```