

ChevoTech

Integrantes:

RM 99404 - Thiago Garcia Tonato

RM 555502 - Ian Madeira Gonçalves da Silva

RM 555109 - Murilo Ribeiro Santos

Sumário

1. Introdução
2. Código Fonte
3. Funcionalidades
4. Introduções de uso
5. Requisitos do sistema
6. Processo de Operação Simples
7. Código Fonte

Introdução do projeto

A solução elaborada foi um conjunto de sensores integrados a um módulo de coleta de dados automotivos. O objetivo deste projeto é oferecer uma solução que permita monitorar diversos parâmetros do veículo, como temperatura, sonda lambda, desgaste dos freios, entre outros, de forma automatizada, eficiente e sem a necessidade de ação humana.

Código Fonte

O código fornecido é escrito em Python e implementa um sistema de coleta e gerenciamento de dados automotivos com base em um conjunto de sensores, permitindo aos usuários interagir com as informações dos sensores e realizar operações relacionadas à manutenção do veículo.

Funcionalidades

Cadastro de usuário: O código permite que novos usuários se cadastrem inserindo um nome de usuário e uma senha. Essas informações são armazenadas nas variáveis `nome` e `senha`. Após o cadastro bem-sucedido, a variável booleana `cadastro` é definida como verdadeira.

Login de Usuário: Após o cadastro, os usuários podem fazer login fornecendo seu nome de usuário e senha. O sistema verifica se as credenciais estão corretas e permite o acesso ao menu principal. Se as credenciais estiverem incorretas ou se o usuário não estiver cadastrado, uma mensagem de login inválido é exibida.

Coleta de Dados dos Sensores: A funcionalidade de coleta de dados dos sensores permite ao usuário simular a obtenção de dados dos sensores. Utilizando a biblioteca `numpy`, são gerados números aleatórios que simulam dados de sensores automotivos. Esses dados são então armazenados em uma lista chamada `dados_sensores`.

Previsão de Manutenção: Com base nos dados coletados dos sensores, o sistema realiza uma previsão sobre a necessidade de manutenção. Calcula-se a média dos dados dos sensores e compara-se com um `threshold` predefinido. Se a média dos dados exceder esse `threshold`, é indicada a necessidade de manutenção. Caso contrário, é exibida uma mensagem informando que nenhuma manutenção é necessária.

Adição da Data de Última Modificação: Esta funcionalidade permite ao usuário adicionar a data da última manutenção realizada. O usuário é solicitado a inserir a data no formato "dd/MM/yyyy", que é então adicionada a uma lista chamada `lista_manutencoes`.

Visualização do Histórico de Manutenções: O sistema permite ao usuário visualizar todas as datas das manutenções previamente adicionadas. Essas datas são exibidas a partir da lista `lista_manutencoes`.

Introduções de Uso

1. Cadastro de Usuário:

- Ao iniciar o programa, escolha a opção "Fazer Cadastro" no menu principal.
- Insira um nome de usuário quando solicitado.
- Insira uma senha quando solicitado.
- Após o cadastro bem-sucedido, você será notificado com a mensagem "Cadastro realizado!".

2. Login de Usuário:

- Escolha a opção "Fazer login" no menu principal.
- Insira o nome de usuário e a senha que você cadastrou anteriormente.
- Se as credenciais estiverem corretas, você será direcionado ao menu principal. Caso contrário, uma mensagem de "Login inválido!" será exibida.

3. Coleta de Dados dos Sensores:

- Após fazer login, escolha a opção "Coletar dados dos sensores" no menu principal.
- Dados dos sensores serão simulados e coletados automaticamente.
- Você receberá uma mensagem indicando que os dados dos sensores foram coletados com sucesso.

4. Previsão de Manutenção:

- Selecione a opção "Prever manutenção" no menu principal.
- O sistema calculará a média dos dados dos sensores coletados.
- Com base nessa média, uma mensagem será exibida indicando se é recomendada uma manutenção ou se nenhuma manutenção é necessária.

5. Adição da Data da Última Manutenção:

- Escolha a opção "Adicionar data da manutenção" no menu principal.
- Insira a data da última manutenção no formato "dd/MM/yyyy" quando solicitado.
- A data será adicionada à lista de manutenções realizadas.

6. Visualização do Histórico de Manutenções:

- Selecione a opção "Histórico de manutenção" no menu principal.
- Todas as datas de manutenção previamente adicionadas serão exibidas.

7. Encerramento do Programa:

- A qualquer momento, escolha a opção "Sair" no menu principal para encerrar o programa.
- Uma mensagem de "LogOut feito! Obrigado!" será exibida antes de encerrar o programa.

Requisitos do sistema

1. Python:

- O sistema requer a instalação do interpretador Python na máquina do usuário.
- Versão recomendada: Python 3.x.

2. Biblioteca Numpy:

- O código faz uso da biblioteca NumPy para geração de números aleatórios simulando dados dos sensores.
- Certifique-se de que a biblioteca NumPy esteja instalada na máquina do usuário.
- Caso não esteja instalada, ela pode ser instalada via pip:

```
pip install numpy
```

3. Biblioteca getpass:

- Essa biblioteca vem automaticamente instalada com o Python.
- Usamos a biblioteca "getpass" para mascarar a senha do usuário ao digitá-la.

4. Biblioteca datetime:

- Essa biblioteca vem automaticamente instalada com o Python.
- Usamos a biblioteca "datetime" para formatar a data inserida pelo usuário.

Processo de Operação Simples

1

Iniciar Programa

O usuário abre o programa e é recebido com um menu de cadastro e login.

2

Cadastro

O usuário se cadastra com login e senha.

3

Login

O usuário inicia login com os dados feitos no cadastro.

4

Menu Principal

Após o login aparece o menu principal com as opções dadas.

Código-fonte

Código-fonte comentado para melhor entendimento do projeto.

```
1 # Importando as bibliotecas necessárias para o funcionamento do código
2 import numpy as np
3 import getpass
4 from datetime import datetime
5
6 # Dicionário para armazenar os usuários e senhas cadastrados
7 usuarios = {}
8 # Lista para armazenar as datas das manutenções realizadas
9 lista_manutencoes = []
10 # Lista para armazenar os dados coletados pelos sensores
11 dados_sensores = []
12
13
14 # Função para cadastrar um novo usuário
15 def Cadastro():
16     login = input("Digite seu login para cadastro: ")
17     senha = getpass.getpass("Digite sua senha para cadastro: ")
18     usuarios[login] = senha
19     print("")
20     print("Cadastro realizado!")
21     return True
22
23
24 # Função para realizar o login do usuário
25 def Login():
26     nome_login = input("Digite seu login: ")
27     senha_login = getpass.getpass("Digite sua senha: ")
28     if nome_login in usuarios and usuarios[nome_login] == senha_login:
29         print("")
30         print("Login realizado com sucesso!")
31         return True
32     else:
33         return False
34
35
36 # Função para simular a coleta de dados dos sensores
37 def ColetarDados():
38     dados_sensores.extend(np.random.rand(10))
39     print("Dados coletados: ", dados_sensores)
40     print("Dados dos sensores coletados com sucesso!")
41
42
43 # Função para prever a necessidade de manutenção com base nos dados coletados
44 def Prever():
45     threshold = 0.45
46     print("")
47     if np.mean(dados_sensores) > threshold:
48         print("Manutenção recomendada.")
49     else:
50         print("Nenhuma manutenção necessária!")
51
52
53 # Função para adicionar a data de uma manutenção à lista de manutenções
54 def AddDataManutencao():
55     print("")
56     while True:
57         manutencao = input("Digite a data da manutenção (dd/MM/yyyy): ")
58         try:
59             data_manutencao = datetime.strptime(manutencao, "%d/%m/%Y")
60             data_formatada = data_manutencao.strftime("%d/%m/%Y")
61             lista_manutencoes.append(data_formatada)
62             print(f"Data de manutenção {data_formatada} adicionada com sucesso!")
63             break
64         except ValueError:
65             print("Data inválida! Digite a data no formato dd/MM/yyyy.")
66
67
68 cadastro = False
69 # Menu com repetição para login
70 while True:
71     print("")
72     print(">>>CHEVOTECH<<<")
73     print("-----")
74     print("1. Fazer Cadastro")
75     print("2. Fazer login")
76     print("3. Sair")
77     print("-----")
78
79     opcao_login = int(input("Digite o número da opção que deseja: "))
80
81     match opcao_login:
82         case 1:
83             # Cadastrar
84             cadastro = Cadastro()
85         case 2:
86             # Login
87             if cadastro:
88                 if Login():
89                     print("")
90                     print(">>>CHEVOTECH<<<")
91                     print("-----")
92                     # Menu principal do programa
93                     while True:
94                         print("")
95                         print(">>>CHEVOTECH<<<")
96                         print("-----")
97                         print("Bem vindo! O que deseja fazer?")
98                         print("")
99                         print("1. Coletar dados dos sensores")
100                        print("2. Prever manutenção")
101                        print("3. Adicionar data da manutenção")
102                        print("4. Histórico de manutenção")
103                        print("5. Sair")
104
105                        opcao_menu = input("Digite o número da opção que deseja: ")
106
107                        match opcao_menu:
108                            case "1":
109                                # Chamando função para coletar dados dos sensores
110                                ColetarDados()
111                            case "2":
112                                # Chamando função para prever a necessidade de manutenção
113                                Prever()
114                            case "3":
115                                # Chamando função para adicionar a data de uma manutenção
116                                AddDataManutencao()
117                            case "4":
118                                # Mostrando o histórico de manutenções realizadas
119                                print("")
120                                print(f"Manutenções feitas nos dias: {lista_manutencoes}")
121                            case "5":
122                                # Finalizando a sessão e saindo do programa
123                                print("")
124                                print("Logout feito! Obrigado!")
125                                break
126                            case _:
127                                # Caso não seja nenhum dos casos anteriores
128                                print("")
129                                print("Digite uma opção válida!")
130
131                        else:
132                            # Se usuário ou senha estiverem incorretos
133                            print("")
134                            print("Login ou senha inválidos!")
135
136                    else:
137                        # Se o usuário tentar fazer login sem ter feito cadastro
138                        print("")
139                        print("Realize o cadastro para fazer login!")
140
141                case 3:
142                    # Sair do programa
143                    print("")
144                    print("Obrigado!")
145                    break
146                case _:
147                    # Caso o usuário digite uma opção inválida
148                    print("Opção inválida")
```

Diferenças entre sprints

Nessa sprint, atualizamos o código para todas as suas funcionalidades serem feitas em funções, modernizando e deixando mais fácil de entender, adicionamos um sistema de mascarar a senha digitada pelo usuário tanto no cadastro quanto no login e adicionamos validação das datas inseridas para que sejam em "dd/MM/yyyy".