

Verificando se Login e senha estão corretos

```
1 import os
2 from http.server import SimpleHTTPRequestHandler
3 import socketserver
4 from urllib.parse import parse_qs
5
6 # Esta versão manin8 verifica se usuário já existe e se senha informada está correta
7 # Em caso de novo usuario, ou seja, um login que não está na base, ele é cadastrado na base e recebe uma mensgaem de boas vinadas
8
9
10 2 usages
11 class MyHandler(SimpleHTTPRequestHandler):
12     1 usage (1 dynamic)
13     def list_directory(self, path):
14         try:
15             # Tenta abrir o arquivo idx.html
16             f = open(os.path.join(path, 'idx.html'), 'r')
17             # Se existir, envia o conteúdo do arquivo
18             self.send_response(200)
19             self.send_header("Content-type", "text/html")
20             self.end_headers()
21             self.wfile.write(f.read().encode('utf-8'))
22             f.close()
23             return None
24         except FileNotFoundError:
25             pass
26
27     return super().list_directory(path)
```

Verificando se Login e senha estão corretos

```
26
27 def do_GET(self):
28     if self.path == '/login':
29         # Tenta abrir o arquivo login.html
30         try:
31             with open(os.path.join(os.getcwd(), 'login.html'), 'r') as login_file:
32                 content = login_file.read()
33                 self.send_response(200)
34                 self.send_header("Content-type", "text/html")
35                 self.end_headers()
36                 self.wfile.write(content.encode('utf-8'))
37         except FileNotFoundError:
38             self.send_error(404, "File not found")
39
40     elif self.path == '/login_failed':...
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57     else:
58         # Se não for a rota "/login", continua com o comportamento padrão
59         super().do_GET()
```

Verificando se Login e senha estão corretos

```
1 usage
61 def usuario_existente(self, login, senha):
62     # Verifica se o login já existe no arquivo
63
64     with open('dados_login.txt', 'r', encoding='utf-8') as file:
65         for line in file:
66             stored_login, stored_senha = line.strip().split(';')
67
68             if login == stored_login:
69                 print("chequei aqui significando que localizei o login informado")
70                 print("senha: " + senha)
71                 print("senha_armazenada: " + senha)
72                 return senha == stored_senha
73     return False
```

Verificando se Login e senha estão corretos

```
74
75     def do_POST(self):
76         # Verifica se a rota é "/enviar_login"
77         if self.path == '/enviar_login':
78             # Obtém o comprimento do corpo da requisição
79             content_length = int(self.headers['Content-Length'])
80             # Lê o corpo da requisição
81             body = self.rfile.read(content_length).decode('utf-8')
82             # Parseia os dados do formulário
83             form_data = parse_qs(body, keep_blank_values=True)
84
85             # Exibe os dados no terminal
86             print("Dados do formulário:")
87             print("Email:", form_data.get('email', [''])[0])
88             print("Senha:", form_data.get('senha', [''])[0])
89
90             # Verifica se o usuário já existe
91             login = form_data.get('email', [''])[0]
92             senha = form_data.get('senha', [''])[0]
93
94             if self.usuario_existente(login, senha):
95                 # Responde ao cliente indicando que o usuário logou com sucesso
96                 self.send_response(200)
97                 self.send_header("Content-type", "text/html; charset=utf-8")
98                 self.end_headers()
99                 mensagem = f"Usuário {login} logado com sucesso!!!"
100                self.wfile.write(mensagem.encode('utf-8'))
```

Verificando se Login e senha estão corretos

```
94         if self.usuario_existente(login, senha):
95             # Responde ao cliente indicando que o usuário logou com sucesso
96             self.send_response(200)
97             self.send_header("Content-type", "text/html; charset=utf-8")
98             self.end_headers()
99             mensagem = f"Usuário {login} logado com sucesso!!!"
100            self.wfile.write(mensagem.encode('utf-8'))
101        else:
102            # Verifica se o login já existe no arquivo
103            if any(line.startswith(f"{login};") for line in
104                  open('dados_login.txt', 'r', encoding='utf-8')):
105                # Redireciona o cliente para a rota "/login_failed"
106                self.send_response(302)
107                self.send_header('Location', '/login_failed')
108                self.end_headers()
109                return # Adicionando um return para evitar a execução do restante do código
110            else:
111                # Adiciona o novo usuário ao arquivo
112                with open('dados_login.txt', 'a', encoding='utf-8') as file:
113                    file.write(f"{login};{senha}\n")
114                # Responde ao cliente com a mensagem de boas-vindas
115                self.send_response(200)
116                self.send_header("Content-type", "text/html; charset=utf-8")
117                self.end_headers()
118                mensagem = f"Olá {login}, seja bem-vindo! Percebemos que você é novo por aqui."
119                self.wfile.write(mensagem.encode('utf-8'))
120        else:
```

Verificando se Login e senha estão corretos

```
109         return # Adicionando um return para evitar a execução do restante do código
110     else:
111         # Adiciona o novo usuário ao arquivo
112         with open('dados_login.txt', 'a', encoding='utf-8') as file:
113             file.write(f"{login};{senha}\n")
114         # Responde ao cliente com a mensagem de boas-vindas
115         self.send_response(200)
116         self.send_header("Content-type", "text/html; charset=utf-8")
117         self.end_headers()
118         mensagem = f"Olá {login}, seja bem-vindo! Percebemos que você é novo por aqui."
119         self.wfile.write(mensagem.encode('utf-8'))
120
121     else:
122         # Se não for a rota "/enviar_login", continua com o comportamento padrão
123         super(MyHandler, self).do_POST()
124
125 # Define o IP e a porta a serem utilizados
126 endereco_ip = "0.0.0.0"
127 porta = 8000
128
129 # Cria um servidor na porta e IP especificados
130 with socketserver.TCPServer((endereco_ip, porta), MyHandler) as httpd:
131     print(f"Servidor iniciado em {endereco_ip}:{porta}")
132     # Mantém o servidor em execução
133     httpd.serve_forever()
134
```