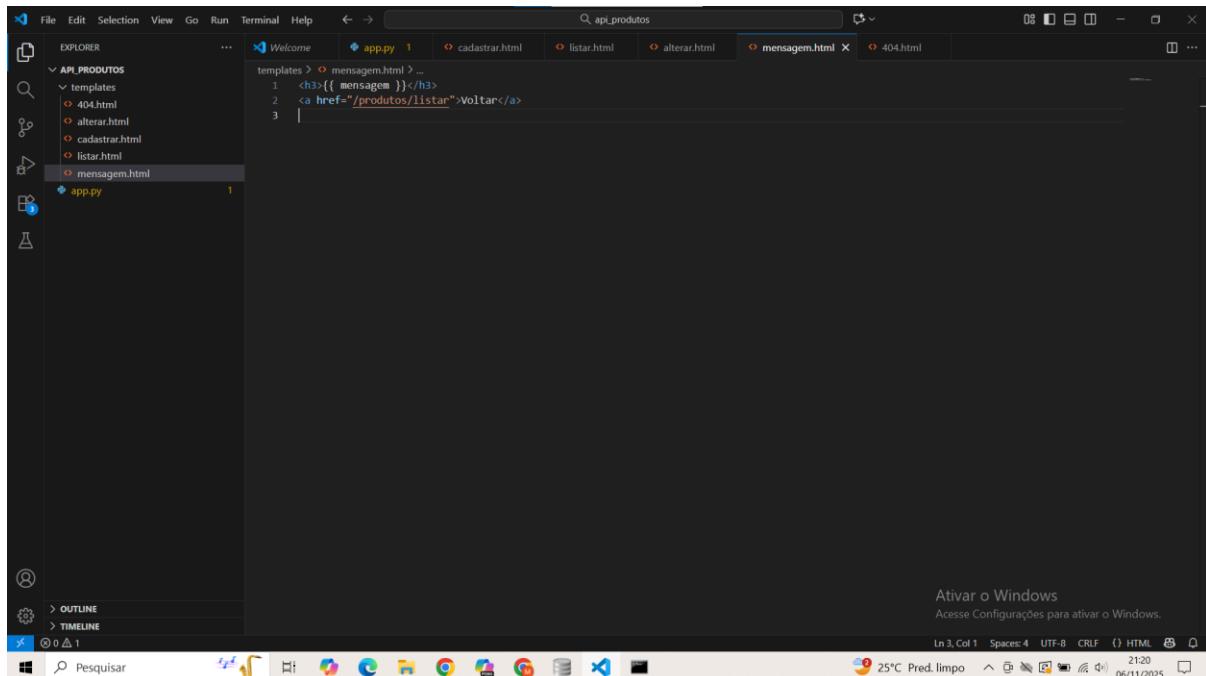


Marina Luiza de Jesus Ferreira 323111931

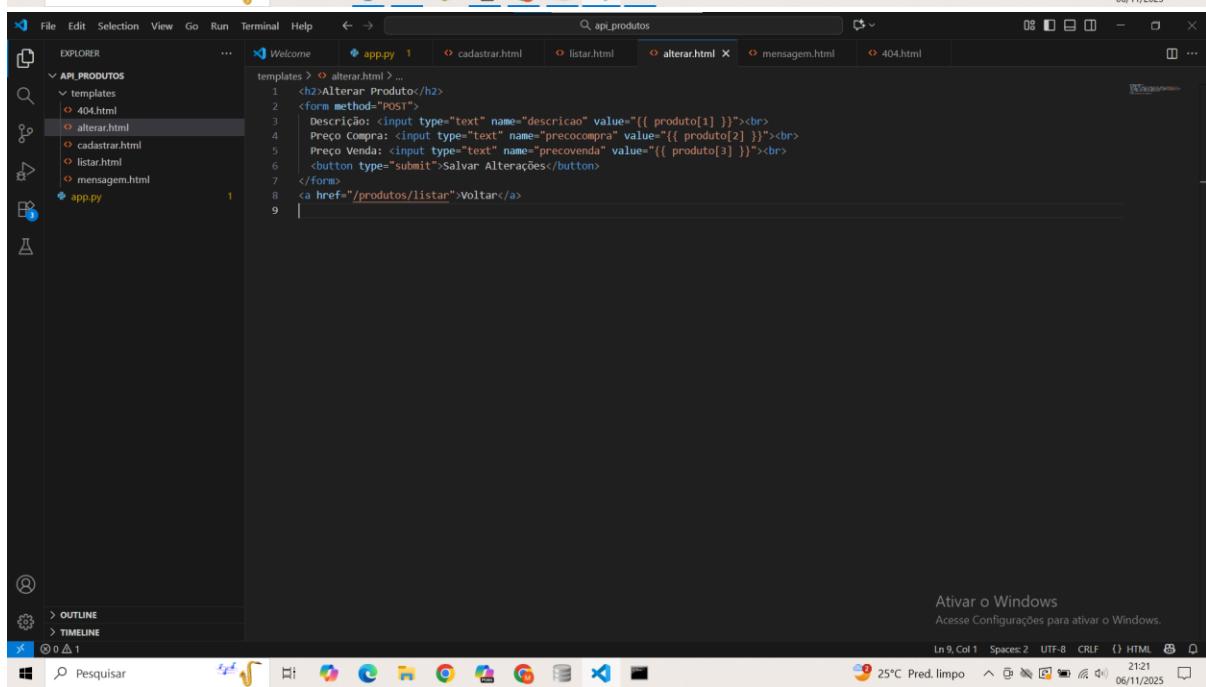
The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows a project structure named "API_PRODUTOS" containing "templates" (with files 404.html, alterar.html, cadastrar.html, listar.html, mensagem.html) and "app.py".
- Terminal:** Shows the command "Welcome" and a list of files: app.py, 404.html, cadastrar.html, listar.html, alterar.html, and mensagem.html.
- Code Editor:** Displays the content of the 404.html template file:

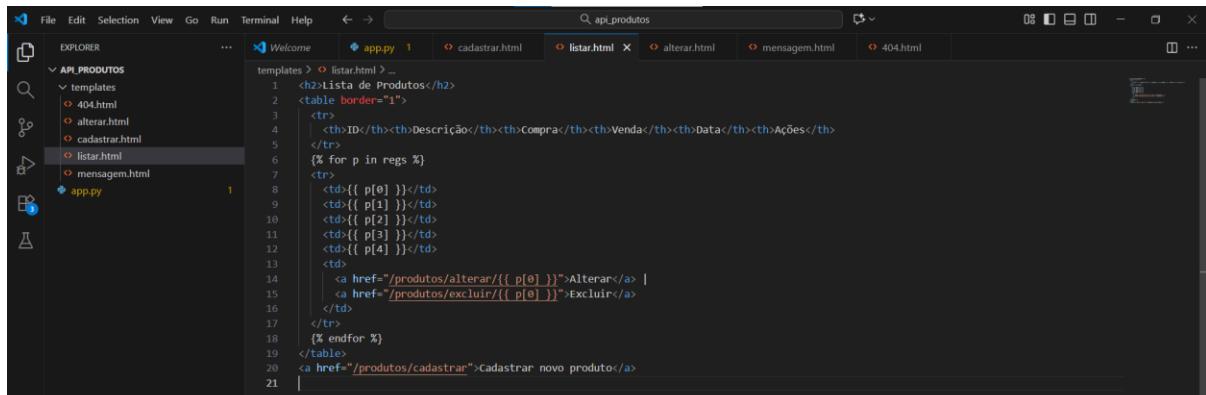
```
templates > 404.html
1 <h1>Página não encontrada</h1>
2 <a href="/produtos/listar">Voltar para lista</a>
```
- Status Bar:** Shows "In 3, Col 1", "Spaces: 4", "UTF-8", "CR LF", "HTML", "21:17", "06/11/2025", and a battery icon.
- Taskbar:** Shows various application icons including File Explorer, Edge, and FileZilla.



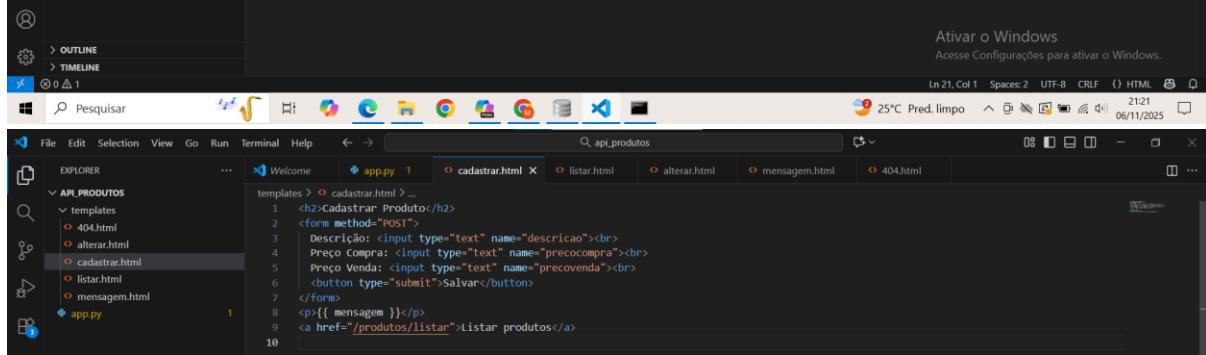
```
templates > mensagem.html > ...
1  <h3>{{ mensagem }}</h3>
2  <a href="/produtos/listar">Voltar</a>
3  |
```



```
templates > alterar.html > ...
1  <h2>Alterar Produto</h2>
2  <form method="POST">
3    Descrição: <input type="text" name="descricao" value="{{ produto[1] }}><br>
4    Preço Compra: <input type="text" name="preco_compra" value="{{ produto[2] }}><br>
5    Preço Venda: <input type="text" name="preco_venda" value="{{ produto[3] }}><br>
6    <button type="submit">Salvar Alterações</button>
7  </form>
8  <a href="/produtos/listar">Voltar</a>
9  |
```



```
templates > listar.html > ...
1  <h2>Lista de Produtos</h2>
2  <table border="1">
3   <tr>
4    <th>ID</th><th>Descrição</th><th>Compra</th><th>Venda</th><th>Data</th><th>Ações</th>
5   </tr>
6   {% for p in regs %}
7    <tr>
8     <td>{{ p[0] }}</td>
9     <td>{{ p[1] }}</td>
10    <td>{{ p[2] }}</td>
11    <td>{{ p[3] }}</td>
12    <td>{{ p[4] }}</td>
13    <td>
14      <a href="/produtos/alterar/{{ p[0] }}">Alterar</a> |
15      <a href="/produtos/excluir/{{ p[0] }}">Excluir</a>
16    </td>
17   </tr>
18  {% endfor %}
19 </table>
20 <a href="/produtos/cadastrar">Cadastrar novo produto</a>
21 |
```



```
templates > cadastrar.html > ...
1  <h2>Cadastrar Produto</h2>
2  <form method="POST">
3   Descrição: <input type="text" name="descricao"><br>
4   Preço Compra: <input type="text" name="precocompra"><br>
5   Preço Venda: <input type="text" name="precovenda"><br>
6   <button type="submit">Salvar</button>
7  </form>
8  <p>{{ mensagem }}</p>
9  <a href="/produtos/listar">Listar produtos</a>
10 |
```



```
templates > cadastrar.html > ...
1  <h2>Cadastrar Produto</h2>
2  <form method="POST">
3   Descrição: <input type="text" name="descricao"><br>
4   Preço Compra: <input type="text" name="precocompra"><br>
5   Preço Venda: <input type="text" name="precovenda"><br>
6   <button type="submit">Salvar</button>
7  </form>
8  <p>{{ mensagem }}</p>
9  <a href="/produtos/listar">Listar produtos</a>
10 |
```

File Edit Selection View Go Run Terminal Help

app.py > cadastrar

```
1 #!/usr/bin/python
2 from datetime import date
3 import sqlite3
4 from sqlite3 import Error
5
6 app = Flask(__name__)
7
8 ##### Cadastrando Produtos
9 @app.route('/produtos/cadastrar', methods=['GET', 'POST'])
10 def cadastrar():
11     if request.method == 'POST':
12         descricao = request.form['descricao']
13         precocompra = request.form['precocompra']
14         precovenda = request.form['precovenda']
15         datacriacao = date.today()
16
17         if descricao and precocompra and precovenda:
18             registro = (descricao, precocompra, precovenda, datacriacao)
19             try:
20                 conn = sqlite3.connect('database/db-produtos.db')
21                 sql = "INSERT INTO produtos(descricao, precocompra, precovenda, datacriacao) VALUES (?, ?, ?, ?)"
22                 cur = conn.cursor()
23                 cur.execute(sql, registro)
24                 conn.commit()
25                 mensagem = 'Produto cadastrado com sucesso!'
26             except Error as e:
27                 mensagem = f'Erro ao cadastrar: {e}'
28             finally:
29                 conn.close()
30
31     return render_template('cadastrar.html', mensagem=mensagem)
32
33 #### Listando Produtos
34 @app.route('/produtos/listar', methods=['GET'])
35 def listar():
36     return render_template('listar.html')
```

File Edit Selection View Go Run Terminal Help

app.py > listar

```
1 #!/usr/bin/python
2 from datetime import date
3 import sqlite3
4 from sqlite3 import Error
5
6 app = Flask(__name__)
7
8 ##### Listando Produtos
9 @app.route('/produtos/listar', methods=['GET'])
10 def listar():
11     try:
12         conn = sqlite3.connect('database/db-produtos.db')
13         sql = "SELECT * FROM produtos"
14         cur = conn.cursor()
15         cur.execute(sql)
16         registros = cur.fetchall()
17         return render_template('listar.html', regs=registros)
18     except Error as e:
19         print(e)
20     finally:
21         conn.close()
22
23 #### Alterando Produto
24 @app.route('/produtos/alterar/int:idproduto', methods=['GET', 'POST'])
25 def alterar(idproduto):
26     conn = sqlite3.connect('database/db-produtos.db')
27     cur = conn.cursor()
28     if request.method == 'POST':
29         descricao = request.form['descricao']
30         precocompra = request.form['precocompra']
31         precovenda = request.form['precovenda']
32         sql = "UPDATE produtos"
33         sql += " SET descricao=?, precocompra=?, precovenda=?"
34         sql += " WHERE idproduto=?"
35         cur.execute(sql, (descricao, precocompra, precovenda, idproduto))
36         conn.commit()
37         conn.close()
38         return render_template('mensagem.html', mensagem='Produto alterado com sucesso!')
39     else:
40         cur.execute("SELECT * FROM produtos WHERE idproduto=?", (idproduto,))
41         produto = cur.fetchone()
42         conn.close()
43         return render_template('alterar.html', produto=produto)
```

The screenshot shows a code editor interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Search Bar:** api_produtos
- Explorer:** API_PRODUTOS folder containing templates (404.html, alterar.html, cadastrar.html, listar.html, mensagem.html) and app.py.
- Code Editor:** The file app.py is open, showing Python code for a web application. The code includes functions for cadastrar, alterar, and excluir products, and handles 404 errors. It uses SQLite for database operations and Jinja2 for templating.
- Status Bar:** Lines 15, Col 47, Spaces: 4, UTF-8, CRLF, Python, 3.12.2 64-bit, São Paulo fraqueja..., 21:22, 06/11/2025.

```
File Edit Selection View Go Run Terminal Help Welcome app.py 1 x cadstrar.html listar.html alterar.html mensagem.html 404.html API_PRODUTOS templates 404.html alterar.html cadastrar.html listar.html mensagem.html app.py 1
54     def alterar(idproduto):
55         ...
56         return render_template('alterar.html', produto=produto)
57
58     #####
59     # 4. Excluir produto
60     @app.route('/produtos/excluir<int:idproduto>')
61     def excluir(idproduto):
62         try:
63             conn = sqlite3.connect('database/db-produtos.db')
64             cur = conn.cursor()
65             cur.execute('DELETE FROM produtos WHERE idproduto=?', (idproduto,))
66             conn.commit()
67             mensagem = 'Produto excluído com sucesso!'
68         except Error as e:
69             mensagem = f'Erro ao excluir: {e}'
70         finally:
71             conn.close()
72         return render_template('mensagem.html', mensagem=mensagem)
73
74     #####
75     # Página de erro 404
76     @app.errorhandler(404)
77     def pagina_nao_encontrada(e):
78         return render_template('404.html'), 404
79
80     #####
81     # Executar a aplicação
82     if __name__ == '__main__':
83         app.run(debug=True)
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
```