

Universidade Federal de Uberlândia
Engenharia de Controle e Automação / Engenharia Mecatrônica
Sistemas Embarcados II / Sistemas Digitais para Mecatrônica
Prof. Éder Alves de Moura
Semana 09 – Servidor Web com Flask



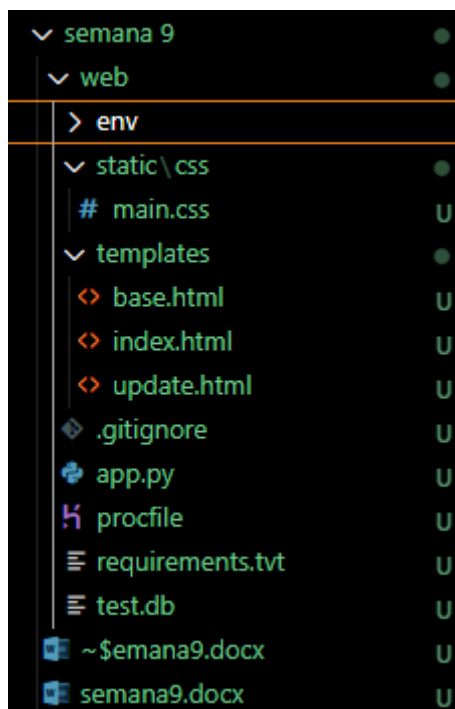
Roteiro de Atividades

Crie uma pasta em seu repositório GitHub, denominada 'Semana09'. Nela você desenvolverá uma aplicação web utilizando o conjunto Python+Flask no Backend (server side).

1. Crie uma subpasta 'web' na pasta 'Semana09' e desenvolva as atividades que estão apresentadas no vídeo:

https://www.youtube.com/watch?v=Z1RJmh_OqeA

Este vídeo apresenta o desenvolvimento de um servidor web com o framework Flask, que utiliza a linguagem Python para a criação de páginas dinâmicas.



Para essa atividade, vamos usar o Python 3.8.2 no Visual Studio Code. Criamos o ambiente de programação com os comandos seguintes:

```
PS D:\Travail\GitHub\SistEmb\Sistemas-Embarcados> cd 'semana 9'
PS D:\Travail\GitHub\SistEmb\Sistemas-Embarcados\semana 9> cd 'web'
PS D:\Travail\GitHub\SistEmb\Sistemas-Embarcados\semana 9\web> pip inst
all virtualenv
```

```
PS D:\Travail\GitHub\SistEmb\Sistemas-Embarcados\semana 9\web> $ virtua  
lenv env
```


```
PS D:\Travail\GitHub\SistEmb\Sistemas-Embarcados\semana 9\web> .\env\Sc  
ripts\activate
```



Depois disso, criamos o arquivo app.py que lança a pagina web criada uma vez criada.

```
semana 9 > web > app.py > update  
1  from flask import Flask, render_template, url_for, request, redirect  
2  from flask_sqlalchemy import SQLAlchemy  
3  from datetime import datetime  
4  
5  app = Flask(__name__)  
6  app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///test.db'  
7  db = SQLAlchemy(app)  
8  
9  class Todo(db.Model):  
10     id = db.Column(db.Integer, primary_key=True)  
11     content = db.Column(db.String(200), nullable=False)  
12     date_created = db.Column(db.DateTime, default=datetime.utcnow)  
13  
14     def __repr__(self):  
15         return '<Task %r>' % self.id  
16  
17  
18  @app.route('/', methods=['POST', 'GET'])  
19  def index():  
20     if request.method == 'POST':  
21         task_content = request.form['content']  
22         new_task = Todo(content=task_content)  
23  
24         try:  
25             db.session.add(new_task)  
26             db.session.commit()  
27             return redirect('/')  
28         except:  
29             return 'There was an issue adding your task'  
30  
31     else:  
32         tasks = Todo.query.order_by(Todo.date_created).all()  
33         return render_template('index.html', tasks=tasks)  
34
```

Finalmente, criamos os arquivos css e html para fazer funcionar o site.

```
semana 9 > web > templates > <> index.html > ...
1  {% extends 'base.html' %}
2
3  {% block head %}
4  <title>Task Master</title>
5  {% endblock %}
6
7  {% block body %}
8  <div class="content">
9      <h1 style="text-align: center">Task Master</h1>
10     {% if tasks|length < 1 %}
11     <h4 style="text-align: center">There are no tasks. Create one below!</h4>
12     {% else %}
13     <table>
14     <tr>
15         <th>Task</th>
16         <th>Added</th>
17         <th>Actions</th>
18     </tr>
19     {% for task in tasks %}
20     <tr>
21         <td>{{ task.content }}</td>
22         <td>{{ task.date_created.date() }}</td>
23         <td>
24             <a href="/delete/{{task.id}}">Delete</a>
25             <br>
26             <a href="/update/{{task.id}}">Update</a>
27         </td>
28     </tr>
29     {% endfor %}
30 </table>
31 {% endif %}
32
33 <div class="form">
34     <form action="/" method="POST">
35         <input type="text" name="content" id="content">
36         <input type="submit" value="Add Task">
37     </form>
38 </div>
39 </div>
40 {% endblock %}
```

semana 9 > web > static > css > # main.css >  #content

```
1  body, html {
2      |   margin: 0;
3      |   font-family: sans-serif;
4      |   background-color:  lightblue;
5  }
6
7  .content {
8      |   margin: 0 auto;
9      |   width: 400px;
10 }
11
12 table, td, th {
13     |   border: 1px solid  #aaa;
14 }
15
16 table {
17     |   border-collapse: collapse;
18     |   width: 100%;
19 }
20
21 th {
22     |   height: 30px;
23 }
24
25 td {
26     |   text-align: center;
27     |   padding: 5px;
28 }
29
30 .form {
31     |   margin-top: 20px;
32 }
33
34 #content {
35     |   width: 70%;
36 }
```

```
semana 9 > web > templates > <> base.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="ie=edge">
7   <link rel="stylesheet" href="{{ url_for('static', filename='css/main.css') }}">
8   {% block head %}{% endblock %}
9 </head>
10 <body>
11   {% block body %}{% endblock %}
12 </body>
13 </html>
```

```
semana 9 > web > templates > <> update.html > ...
1 {% extends 'base.html' %}
2
3 {% block head %}
4 <title>Task Master</title>
5 {% endblock %}
6
7 {% block body %}
8 <div class="content">
9   <h1 style="text-align: center;">Update Task</h1>
10
11   <div class="form">
12     <form action="/update/{{task.id}}" method="POST">
13       <input type="text" name="content" id="content" value="{{task.content}}">
14       <input type="submit" value="Update">
15     </form>
16   </div>
17 </div>
18 {% endblock %}
```

Obtemos o resultado seguinte:

Task Master

Task	Added	Actions
do the dishes	2019-05-20	Delete Update

Podemos criar tarefas, agendar a data e remover elas.