09

Mãos na massa: Persistindo com MySQL

> Caso você queira o projeto completo do primeiro curso, pode baixá-lo aqui (https://caelum-online-public.s3.amazonaws.com/694-flask-rotas-templates-autenticacao/06/aula6.zip).

Atenção: Antes de começar a fazer o exercício você precisa ter um MySQL instalado na sua máquina ou rodando em algum local acessível. Caso nunca tenha usado MySQL com Python, execute os seguintes comandos para configurar seu sistema:

MacOS

```
brew install mysql
pip3 install mysqlclient
```

Ubuntu

```
sudo apt-get install libmysqlclient-dev
pip3 install mysqlclient
```

Windows

```
pip3.exe install mysqlclient
```

Chegou a hora de você executar o que foi visto na aula! Para isso, execute os passos listados abaixo.

---

1) Para realizar a conexão com o MySQL, utilizando o Flask, instale a extensão **Flask-MySQLdb**. Na pasta da aplicação **jogoteca**, execute:

```
pip3 install flask_mysqldb==0.2.0
```

2) Faça o download dos arquivos **prepara_banco.py** e **dao.py** aqui (https://s3.amazonaws.com/caelum-online-public/739-python-flask2/01/download.zip). Extraia o zip e copie os arquivos para dentro do projeto **jogoteca**. Execute o arquivo **prepara_banco.py** para criar o seu banco de dados e inserir alguns dados:

> Lembre-se de alterar as informações de usuário, senha, host e porta deste arquivo de acordo com o seu banco de dados.

```
python3 prepara_banco.py
```

3) Já no projeto, dentro de **jogoteca.py**, importe a classe `JogoDao` do módulo `dao` :

```
from dao import JogoDao
```

4) Importe também a classe `MySQL` , do pacote `flask_mysqldb` :

```
from flask_mysqldb import MySQL
```

5) Agora, antes da criação da classe `Jogo` , configure o acesso ao banco de dados:

Lembre-se de mudar as configurações de acordo com o seu banco!

```
...
app = Flask(__name__)
app.secret_key = 'alura'

app.config['MYSQL_HOST'] = "0.0.0.0"
app.config['MYSQL_USER'] = "root"
app.config['MYSQL_PASSWORD'] = "admin"
app.config['MYSQL_DB'] = "jogoteca"
app.config['MYSQL_PORT'] = 3306

class Jogo:
...
```

E instancie-o em seguida:

```
db = MySQL(app)
```

6) Com o banco de dados instanciado, instancie um `JogoDao` , passando para ele o banco de dados:

```
jogo_dao = JogoDao(db)
```

7) Modifique a função `criar` , para que fique desta forma:

```
@app.route('/criar', methods=['POST',])
def criar():
    nome = request. form['nome']
    categoria = request. form['categoria']
    console = request. form['console']
    jogo = Jogo(nome, categoria, console)
    jogo_dao.salvar(jogo)
    return redirect(url_for('index'))
```

8) Vamos extrair as classes `Jogo` e `Usuario` para um novo módulo. Basta criar este arquivo **models.py** e colar as duas classes.

9) Como as classes foram extraídas, importe-as em **jogoteca.py**:

```
from models import Jogo, Usuario
```

E não esqueça de importá-las também em **dao.py**, apagando o `import` antigo.

10) Na classe `JogoDao`, o método `salvar` precisa do `id` do jogo, então crie-o na classe `Jogo`, tornando-o opcional através do valor `None`, desta forma não quebramos o nosso código:

```
class Jogo:
    def __init__(self, nome, categoria, console, id=None):
        self.id = id
        self.nome = nome
        self.categoria = categoria
        self.console = console
```

11) Ainda em **jogoteca.py**, remova a lista antiga, que estava sendo criada na mão, e na função `index`, crie uma lista chamando o método `listar` de `JogoDao`:

```
@app.route('/')
def index():
    lista = jogo_dao.listar()
    return render_template('lista.html', titulo='Jogos', jogos=lista)
```

12) Para a questão do usuário, remova os usuários que estão sendo criados na mão, importe o `UsuarioDao` do módulo `dao` e instancie-o, passando para ele o banco de dados:

```
usuario_dao = UsuarioDao(db)
```

13) E na função `autenticar`, não há mais necessidade de verificar se o usuário está na lista de usuários que você acabou de remover. No lugar dessa verificação, busque o usuário a partir do método `buscar_por_id`, do `UsuarioDao`, passando como parâmetro para ele o usuário da requisição. Por fim, execute o login do usuário caso o usuário exista:

```
@app.route('/autenticar', methods=['POST',])
def autenticar():
    usuario = usuario_dao.buscar_por_id(request.form['usuario'])
    if usuario:
        if usuario.senha == request.form['senha']:
            session['usuario_logado'] = usuario.id
            flash(usuario.nome + ' logou com sucesso!')
            proxima_pagina = request.form['proxima']
            return redirect(proxima_pagina)
    else:
        flash('Não logado, tente denovo!')
        return redirect(url_for('login'))
```