PYSTACK WEEK 4.0 | AULA 01

Acesse o material diretamente pelo Notion, segue o link:

```
https://grizzly-amaranthus-f6a.notion.site/PYSTACK-WEEK-4-0-AULA-01-5a288e9287584eceb44482dc844f2244
```

Avisos

Para tirar dúvidas acesse nossa comunidade do Discord

```
https://pythonando.com.br/inscricao/links
```

Não esqueça de responder os desafios:

```
https://pythonando.com.br/evento/desafios
```

Conteúdo técnico

Primeiro vamos criar e ativar o ambiente virtual:

```
+# Criar
  # Linux
    python3 -m venv venv
  # Windows
    python -m venv venv

#Ativar
  # Linux
    source venv/bin/activate
  # Windows
    venv/Scripts/Activate

# Caso algum comando retorne um erro de permissão execute o código e tente novamente:

  Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned
```

Instale as bibliotecas:

```
pip install django
pip install pillow
```

Inicio um projeto Django

```
django-admin startproject nutri_lab .
```

Configure os arquivos estáticos:

```
STATIC_URL = '/static/'
STATICFILES_DIRS = (os.path.join(BASE_DIR, 'templates/static'),)
STATIC_ROOT = os.path.join('static')

MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'
```

Crie um app autenticacão:

```
python3 manage.py startapp autenticacao
```

Crie uma URL para nosso novo APP:

```
path('auth/', include('autenticacao.urls')),
```

Crie uma URL para login e cadastro:

```
path('cadastro/', views.cadastro, name='cadastro'),
path('logar/', views.logar, name='logar'),
```

Crie as views:

```
def cadastro(request):
    return render(request, 'cadastro.html')

def logar(request):
    pass
```

Crie o arquivo cadastro.html

Crie um arquivo base.html

```
{% load static %}
<!doctype html>
<html lang="pt-BR">
  <head>
    {% block 'head' %}{% endblock  %}
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">
```

```
    <title>{% block 'titulo' %}{% endblock  %}</title>
  </head>
  <body>


    {% block 'body' %}{% endblock  %}


    <script src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js" ></script>


  </body>
</html>
```

Desenvolva o cadastro.html

```
{% extends 'base.html' %}

{% block 'titulo' %}Cadastre-se{% endblock %}

{% block 'body' %}
    <div class="container-fluid">
        <div class="row">
            <div class="col-md-3 sidebar">
                <div class="logo_sidebar">

                    <h2 class="logo">
                        <img class="img_logo_sidebar" src="">
                    </h2>

                </div>


            </div>

            <div class="col-md-9">
                <div class="area_cadastrar">
                    <h2 class="titulo">Seja bem vindo</h2>
                    <p class="sub-titulo">Crie sua conta</p>
                    <hr>

                    <form action="" method="">
                        <span class="bold">Nome de usuário</span>
                        <input class="form-control input-form" name="usuario" type="text" placeholder="Digite seu nome...">
                        <br>
                        <span class="bold">E-mail</span>
                        <input class="form-control input-form" name="email" type="text" placeholder="Digite seu E-mail...">
                        <br>
                        <div class="row">
                            <div class="col-md">
                                <span class="bold">Senha</span>
                                <input class="form-control input-form" name="senha" type="password" placeholder="Digite sua senha...">
                            </div>

                            <div class="col-md">
                                <span class="bold">Confirmar senha</span>
                                <input class="form-control input-form" name="confirmar_senha" type="password" placeholder="Confirme sua sen
                            </div>
                        </div>
                        <br>
                        <button class="btn-first">Registrar</button>

                    </form>

                    <h3 style="position: absolute; bottom: 65px; font-weight: bold">Já possui uma conta ?  <a class="btn btn-outline-su
                </div>

            </div>

        </div>


    </div>
```

```
{% endblock %}
```

Agora crie o arquivo cadastro.css

```
.sidebar{
    background-color: #27DB8F;
    height: 100vh;
}


.logo_sidebar{
    text-align: center;
}

.img_logo_sidebar{
    width: 100%;


}

.sidebar{

    background-image: url();

}

.area_cadastrar{
    margin-top: 8%;
    margin-left: 15%;
    width: 70%;
}

.titulo{

    font-weight: bold;

}

.sub-titulo{
    color: gray;
}

.input-form{
    margin-top: 15px;
    width: 100%;
    padding: 5px;
    outline: 0;

}

.input-form:focus{

    border: 2px solid #27DB8F !important;
    outline: 0 !important;
    box-shadow: none !important;

}

.bold{
    font-weight: bold;
}

.btn-first{

    border: none;
    padding: 5px 15px 5px 15px;
    font-size: 25px;
    background-color: #27DB8F;
    color: white;
    font-weight: bold;

}
```

Importe o arquivo CSS no HTML

```
{% load static %}

{% block 'head' %}
    <link rel="stylesheet" href="{% static 'autenticacao/css/cadastro.css' %}">

{% endblock %}
```

Adicione o ACTION E METHOD no FORM

```
<form action="{% url 'cadastro' %}" method="POST">
```

Altere a view para diferenciar POST e GET:

```
def cadastro(request):
    if request.method == "GET":
        return render(request, 'cadastro.html')
    elif request.method == "POST":
        username = request.POST.get('usuario')
        senha = request.POST.get('senha')
        email = request.POST.get('email')
        confirmar_senha = request.POST.get('confirmar_senha')

        return HttpResponse(confirmar_senha)
```

DICA PARA TREINAR: TENTE VALIDAR SE O USERNAME, EMAIL E SENHA NÃO FORAM ENVIADOS EM BRANCO

Faça as validações:

```
import re
from django.contrib import messages
from django.contrib.messages import constants

def password_is_valid(request, password, confirm_password):
    if len(password) < 6:
        messages.add_message(request, constants.ERROR, 'Sua senha deve conter 6 ou mais caractertes')
        return False

    if not password == confirm_password:
        messages.add_message(request, constants.ERROR, 'As senhas não coincidem!')
        return False

    if not re.search('[A-Z]', password):
        messages.add_message(request, constants.ERROR, 'Sua senha não contem letras maiúsculas')
        return False

    if not re.search('[a-z]', password):
        messages.add_message(request, constants.ERROR, 'Sua senha não contem letras minúsculas')
        return False

    if not re.search('[1-9]', password):
        messages.add_message(request, constants.ERROR, 'Sua senha não contém números')
        return False

    return True
```

Adicione nas views.py

```
if not password_is_valid(request, senha, confirmar_senha):
        return redirect('/auth/cadastro')
```

Salve os dados do novo usuário no banco:

Leia aqui! Caso tente cadastrar usuários com username já existente irá dar erro, pois não validamos esse caso. Ficou de lição de casa para tentar realizar essa validação.

```
try:
    user = User.objects.create_user(username=username,
                                    email=email,
                                    password=senha,
                                    is_active=False)
    user.save()
    return redirect('/auth/logar')
except:
    return redirect('/auth/cadastro')
```

Vamos agora preparar as mensagens de erro:

Primeiro fazemos as configurações no settings.py

```
from django.contrib.messages import constants

MESSAGE_TAGS = {
    constants.DEBUG: 'alert-primary',
    constants.ERROR: 'alert-danger',
    constants.SUCCESS: 'alert-success',
    constants.INFO: 'alert-info',
    constants.WARNING: 'alert-warning',
}
```

Adicione as mensagens onde achar necessário:

```
from django.contrib import messages
from django.contrib.messages import constants

messages.add_message(request, constants.ERROR, 'Sua mensagem aqui')
```

Exiba as mensagens no template HTML:

```
{% if messages %}
        <br>
        {% for message in messages %}
            <div class="alert {{message.tags}}">
                {{message}}
            </div>
        {% endfor %}
{% endif %}
```

Agora vamos para o login

Crie o arquivo login.html:

```
{% extends 'base.html' %}
{% load static %}

{% block 'head' %}
    <link rel="stylesheet" href="{% static 'autenticacao/css/css.css' %}">
{% endblock %}

{% block 'titulo' %}Cadastre-se{% endblock %}

{% block 'body' %}
    <div class="container-fluid">

        <div class="row">
            <div class="col-md-3 sidebar">
                <div class="logo_sidebar">

                    <h2 class="logo">
                        <img class="img_logo_sidebar" src="{% static 'autenticacao/img/logo_nutri_lab.png' %}">
                    </h2>

                </div>


            </div>

            <div class="col-md-9">
                <div class="area_cadastrar">
                    {% if messages %}
                        <br>
                        {% for message in messages %}
                            <div class="alert {{message.tags}}">
                                {{message}}
                            </div>
                        {% endfor %}
                    {% endif %}
                    <h2 class="titulo">Seja bem vindo</h2>
                    <p class="sub-titulo">Entre com sua conta</p>
                    <hr>

                    <form action="" method="POST">{% csrf_token %}
                        <span class="bold">Nome de usuário</span>
                        <input class="form-control input-form" name="usuario" type="text" placeholder="Digite seu nome...">
                        <br>
                        <span class="bold">Senha</span>
                        <input class="form-control input-form" name="senha" type="password" placeholder="Digite sua senha...">
                        <br>
                        <button class="btn-first">Logar</button>

                    </form>

                    <h3 style="position: absolute; bottom: 65px; font-weight: bold">Não tem uma conta ?  <a class="btn btn-outline-succ
                </div>

            </div>

        </div>


    </div>



{% endblock %}
```

Crie a view logar para realizar a autenticação do usuário:

```
def logar(request):
    if request.method == "GET":
        return render(request, 'logar.html')
    elif request.method == "POST":
        username = request.POST.get('usuario')
        senha = request.POST.get('senha')
```

```
        usuario = auth.authenticate(username=username, password=senha)

        if not usuario:
            messages.add_message(request, constants.ERROR, 'Username ou senha inválidos')
            return redirect('/auth/logar')
        else:
            auth.login(request, usuario)
            return redirect('/')
```

Por fim é só verificar se o usuários já está logado antes de exibir logar e cadastro:

```
if request.user.is_authenticated:
        return redirect('/')
```

Para realizar o logout do usuário cria uma URL para sair:

```
path('sair/', views.sair, name="sair")
```

Crie a view para sair:

```
def sair(request):
    auth.logout(request)
    return redirect('/auth/logar')
```

Vamos enviar o E-mail de cadastro confirmado

Configure em settings.py

```
# Email

EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
EMAIL_HOST_USER = "nome@email.com.br"
```

Crie a função para enviar E-mail

```
from django.core.mail import EmailMultiAlternatives
from django.template.loader import render_to_string
from django.utils.html import strip_tags
from django.conf import settings

def email_html(path_template: str, assunto: str, para: list, **kwargs) -> dict:

    html_content = render_to_string(path_template, kwargs)
    text_content = strip_tags(html_content)

    email = EmailMultiAlternatives(assunto, text_content, settings.EMAIL_HOST_USER, para)

    email.attach_alternative(html_content, "text/html")
    email.send()
    return {'status': 1}
```

Crie o template do E-mail

```
<html>
    <head>

    </head>


    <body>

        <h1>Seja bem vindo {{username}}, sua conta foi criada com sucesso.</h1>
        <span><a href="{{link_ativacao}}">CLIQUE AQUI</a>ativar sua conta.</span>

    </body>


</html>
```

Envie o E-mail após o cadastro

```
import os
from django.conf import settings

path_template = os.path.join(settings.BASE_DIR, 'autenticacao/templates/emails/cadastro_confirmado.html')
email_html(path_template, 'Cadastro confirmado', [email,], username=username)
```

Crie a models

```
class Ativacao(models.Model):
    token = models.CharField(max_length=64)
    user = models.ForeignKey(User, on_delete=models.DO_NOTHING)
    ativo = models.BooleanField(default=False)

    def __str__(self):
        return self.user.username
```

Faça as migrações e inclua no admin.py

Crie a URL

```
path('ativar_conta/<str:token>/', views.ativar_conta, name="ativar_conta")
```

Crie a views para ativar a conta

```
def ativar_conta(request, token):
    token = get_object_or_404(Ativacao, token=token)
    if token.ativo:
        messages.add_message(request, constants.WARNING, 'Essa token já foi usado')
        return redirect('/auth/logar')
    user = User.objects.get(username=token.user.username)
    user.is_active = True
    user.save()
    token.ativo = True
    token.save()
    messages.add_message(request, constants.SUCCESS, 'Conta ativa com sucesso')
    return redirect('/auth/logar')
```

Envie o link por E-mail

```
email_html(path_template, 'Cadastro confirmado', [email,], username=username, link_ativacao="127.0.0.1:8000/auth/ativar_conta/{token}")
```

E com isso finalizamos nossa primeira aula da PSW 4.0

Assets NUTRI LAB

https://drive.google.com/drive/folders/1EaEbfu4XX9zVOTbNa83EpSYzUuvywRka?usp=sharing