AULA 3

Acesse o material direto pelo Notion!

```
https://grizzly-amaranthus-f6a.notion.site/AULA-3-f5c5f427ae8f4eddbe5ba87d4ef4033f?pvs=4
```

▼ Relatório

Crie uma URL para os relatórios:

```
path('relatorio/<int:id>/', views.relatorio, name='relatorio')
```

Crie a view para os relatórios:

```
def relatorio(request, id):
    desafio = Desafio.objects.get(id=id)

    return render(request, 'relatorio.html', {'desafio': desafio},)
```

Crie o HTML:

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

{% block 'cabecalho' %}

    <link href="{% static 'usuarios/css/cadastro.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/novo_flashcard.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/iniciar_desafio.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/desafio.css' %}" rel="stylesheet">
    <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

{% endblock 'cabecalho' %}


{% block 'conteudo' %}
    {% include "partials/header.html" %}
    <div class="container">
        <br>
        <br>
        <div class="row">
            <div class="col-md">
                <p class="fonte-secundaria">{{desafio.titulo}}</p>
                <hr>
                <canvas id="grafico1"></canvas>
            </div>
            <div class="col-md">
                <br><br><br><br>

                <div style="background-color: white;">

                    <canvas id="grafico2"></canvas>
                </div>
            </div>
        </div>

    </div>

{% endblock 'conteudo' %}
```

Em desafio.html redirecione para o relatório:

```
<a class="btn-cadastro btn-desafio" href="{% url 'relatorio' desafio.id %}">Relatório detalhado</a>
```

Importe o Chart.js

```
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
```

Adicione o JS para os gráficos:

```
<script>
        const ctx = document.getElementById('grafico1');

        new Chart(ctx, {
          type: 'pie',
          data: {
            labels: ['Acertos', 'Erros'],
            datasets: [{
              label: 'Qtd',
              data: [10, 5],
              borderWidth: 1
            }]
          },

        });
    </script>

    <script>

        const ctx2 = document.getElementById('grafico2');
```

```
        new Chart(ctx2, {
          type: 'radar',
          data: {
            labels: ['Matemática', 'Português', 'Programação', 'História'],
            datasets: [{
              label: 'Qtd',
              data: [2, 4, 8, 10],
              borderWidth: 1,
              fill: true,
              backgroundColor: 'rgba(255, 99, 132, 0.2)',
              borderColor: 'rgb(255, 99, 132)',
              pointBackgroundColor: 'rgb(255, 99, 132)',
              pointBorderColor: '#fff',
              pointHoverBackgroundColor: '#fff',
              pointHoverBorderColor: 'rgb(255, 99, 132)'
            }]
          },

        });

    </script>
```

Agora, precisamos deixar os gráficos dinâmicos, para isso vamos calcular os acertos e erros na view:

```
def relatorio(request, id):
    desafio = Desafio.objects.get(id=id)


    acertos = desafio.flashcards.filter(acertou=True).count()
    erros = desafio.flashcards.filter(acertou=False).count()

    dados = [acertos, erros]

    return render(request, 'relatorio.html', {'desafio': desafio, 'dados': dados},)
```

Altere no script para o valor de dados:

```
data: {{dados}},
```

Agora vamos buscar todas as categorias dos desafios e a quantidade de acerto por categoria:

```
def relatorio(request, id):
    desafio = Desafio.objects.get(id=id)


    acertos = desafio.flashcards.filter(acertou=True).count()
    erros = desafio.flashcards.filter(acertou=False).count()

    dados = [acertos, erros]


    categorias = desafio.categoria.all()
    name_categoria = [i.nome for i in categorias]

    dados2 = []
    for categoria in categorias:
        dados2.append(desafio.flashcards.filter(flashcard__categoria=categoria).filter(acertou=True).count())

    return render(request, 'relatorio.html', {'desafio': desafio, 'dados': dados, 'categorias': name_categoria, 'dados2': dados2,},)
```

Agora insira os dados no script:

```
new Chart(ctx2, {
    type: 'radar',
    data: {
      labels: {{categorias|safe}},
      datasets: [{
        label: 'Qtd',
        data: {{dados2}},
        borderWidth: 1,
        fill: true,
        backgroundColor: 'rgba(255, 99, 132, 0.2)',
        borderColor: 'rgb(255, 99, 132)',
        pointBackgroundColor: 'rgb(255, 99, 132)',
        pointBorderColor: '#fff',
        pointHoverBackgroundColor: '#fff',
        pointHoverBorderColor: 'rgb(255, 99, 132)'
      }]
    },

});
```

▼ Nova apostila

Crie um novo app:

```
python3 manage.py startapp apostilas
```

Instale o APP!!

Crie uma URL para as apostilas:

```
path('apostilas/', include('apostilas.urls')),
```

Crie a model Apostila:

```
class Apostila(models.Model):
    user = models.ForeignKey(User, on_delete=models.DO_NOTHING)
    titulo = models.CharField(max_length=100)
    arquivo = models.FileField(upload_to='apostilas')


    def __str__(self):
        return self.titulo
```

Faça as migrações!!

Cria uma URL para criar novas apostilas:

```
from django.urls import path
from . import views

urlpatterns = [
    path( 'adicionar_apostilas/', views.adicionar_apostilas, name='adicionar_apostilas'),
]
```

Crie a view:

```
def adicionar_apostilas(request):
    if request.method == 'GET':
        return render(request, 'adicionar_apostilas.html')
```

Crie o HTML:

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

{% block 'cabecalho' %}
```

```
    <link href="{% static 'usuarios/css/cadastro.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/novo_flashcard.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/iniciar_desafio.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/desafio.css' %}" rel="stylesheet">
    <link href="{% static 'apostilas/css/adicionar_apostilas.css' %}" rel="stylesheet">
    <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

{% endblock 'cabecalho' %}


{% block 'conteudo' %}
    {% include "partials/header.html" %}
    <div class="container">
        <br><br>
        <div class="row">
            <div class="col-md">

                <div class="box-form">
                    {% if messages %}
                        <br>
                        {% for message in messages %}
                            <section class="alert {{message.tags}}">
                                {{message}}
                            </section>
                        {% endfor %}
                        <br>
                    {% endif %}
                    <form action="" method="POST" enctype='multipart/form-data'>{% csrf_token %}
                        <p class="fonte-secundaria">Adicionar resumos</p>
                        <label>Titulo</label>
                        <input type="text" name="titulo" placeholder="titulo" class="form-control">
                        <br>
                        <label>Arquivo</label>
                        <input type="file" name="arquivo" class="form-control">
                        <br>
                        <input type="submit" class="btn-cadastro" value="Enviar">
                    </form>
                </div>

            </div>
            <div class="col-md">

                <table>
                    <thead>
                      <tr>
                        <th scope="col">Titulo</th>
                        <th scope="col">Ação</th>
                      </tr>
                    </thead>
                    <tbody>


                            <tr class="linha">
                                <td>Titulo</td>
                                <td><a href="" style="color: black; text-decoration: none;" class="btn-cadastro">Abrir</a></td>
                            </tr>

                    </tbody>
                </table>
                <hr>
                <p class="fonte-secundaria">Views totais: </p>
            </div>
        </div>
    </div>
{% endblock 'conteudo' %}
```

Receba e salve o PDF:

```
def adicionar_apostilas(request):
    if request.method == 'GET':
        return render(
            request, 'adicionar_apostilas.html'
        )
    elif request.method == 'POST':
        titulo = request.POST.get('titulo')
        arquivo = request.FILES['arquivo']

        apostila = Apostila(user=request.user, titulo=titulo, arquivo=arquivo)
        apostila.save()
        messages.add_message(
            request, constants.SUCCESS, 'Apostila adicionada com sucesso.'
        )
        return redirect('/apostilas/adicionar_apostilas/')
```

Adicione uma URL para os arquivos de media:

```
from django.contrib import admin
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('usuarios/', include('usuarios.urls')),
    path('flashcard/', include('flashcard.urls')),
    path('apostilas/', include('apostilas.urls')),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Para listar as apostilas busque as do usuário na view e envie ao HTML:

```
def adicionar_apostilas(request):
    if request.method == 'GET':
        apostilas = Apostila.objects.filter(user=request.user)
        return render(request, 'adicionar_apostilas.html', {'apostilas': apostilas})
```

Liste as apostilas no HTML:

```
{% for apostila in apostilas %}
    <tr class="linha">
        <td>{{apostila.titulo}}</td>
        <td><a href="#" style="color: black; text-decoration: none;" class="btn-cadastro">Abrir</a></td>
    </tr>
{% endfor %}
```

Crie a model ViewApostila e execute as migrações!!

```
class ViewApostila(models.Model):
    ip = models.GenericIPAddressField()
    apostila = models.ForeignKey(Apostila, on_delete=models.DO_NOTHING)

    def __str__(self):
        return self.ip
```

Busque a quantidade de views que teve as apostilas de um usuário:

```
views_totais = ViewApostila.objects.filter(apostila__user = request.user).count()
```

Envie para o HTML e exiba:

```
<p class="fonte-secundaria">Views totais: {{views_totais}}</p>
```

▼ Apostila

Cria a view apostila:

```
def apostila(request, id):
    apostila = Apostila.objects.get(id=id)
```

```
    return render(request, 'apostila.html', {'apostila': apostila})
```

Crie a URL para apostila:

```
path('apostila/<int:id>', views.apostila, name='apostila'),
```

Crie o HTML de apostilas:

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

{% block 'cabecalho' %}

    <link href="{% static 'usuarios/css/cadastro.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/novo_flashcard.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/iniciar_desafio.css' %}" rel="stylesheet">
    <link href="{% static 'flashcard/css/desafio.css' %}" rel="stylesheet">
    <link href="{% static 'apostilas/css/adicionar_apostilas.css' %}" rel="stylesheet">
    <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

{% endblock 'cabecalho' %}


{% block 'conteudo' %}
    {% include "partials/header.html" %}
    <div class="container">
        <br><br>
        <div class="row">
            <div class="col-md">

                <a href="{{apostila.arquivo.url}}"><div style="width: 100%;" id="pdf-container">Teste</div></a>

            </div>
            <div class="col-md">

                <p>Views únicas:</p>
                <p>Views totais: </p>
            </div>
        </div>
    </div>

{% endblock 'conteudo' %}
```

Em adicionar apostila redirecione para as apostilas:

```
<td><a href="{% url "apostila" apostila.id %}" style="color: black; text-decoration: none;" class="btn-cadastro">Abrir</a></td>
```

Exiba o PDF no HTML com JS:

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/pdf.js/2.11.338/pdf.min.js"></script>
<script>

    const container = document.getElementById('pdf-container');

    pdfjsLib.getDocument("{{apostila.arquivo.url}}").promise.then(pdf => {
        pdf.getPage(1).then(page => {
            const canvas = document.createElement('canvas');
            const context = canvas.getContext('2d');
            const viewport = page.getViewport({ scale: 0.6 });

            canvas.width = viewport.width;
            canvas.height = viewport.height;

            page.render({ canvasContext: context, viewport }).promise.then(() => {
                container.appendChild(canvas);
            });
        });
    });
</script>
```

Quando um usuário acessar uma apostila contabilize uma View:

```
view = ViewApostila(
    ip=request.META['REMOTE_ADDR'],
    apostila=apostila
)
view.save()
```

Busque as views únicas e totais e envie para o HTML:

```
views_unicas = ViewApostila.objects.filter(apostila=apostila).values('ip').distinct().count()
views_totais = ViewApostila.objects.filter(apostila=apostila).count()
```

Exiba no HTML o resultado:

```
<p>Views únicas: {{views_unicas}}</p>
<p>Views totais: {{views_totais}}</p>
```

▼ Python Full

Conheça nosso curso completo de Python e Django que te da acesso à:

- Mais de 630 aulas
- Agendamento de reuniões com professores
- Análises de códigos
- Eventos entre alunos
- Exercícios automáticos
- E muito mais

Para quem participou da PSW 9.0 terá um desconto especial, confira no link abaixo:

https://youtu.be/DMQSziesTko