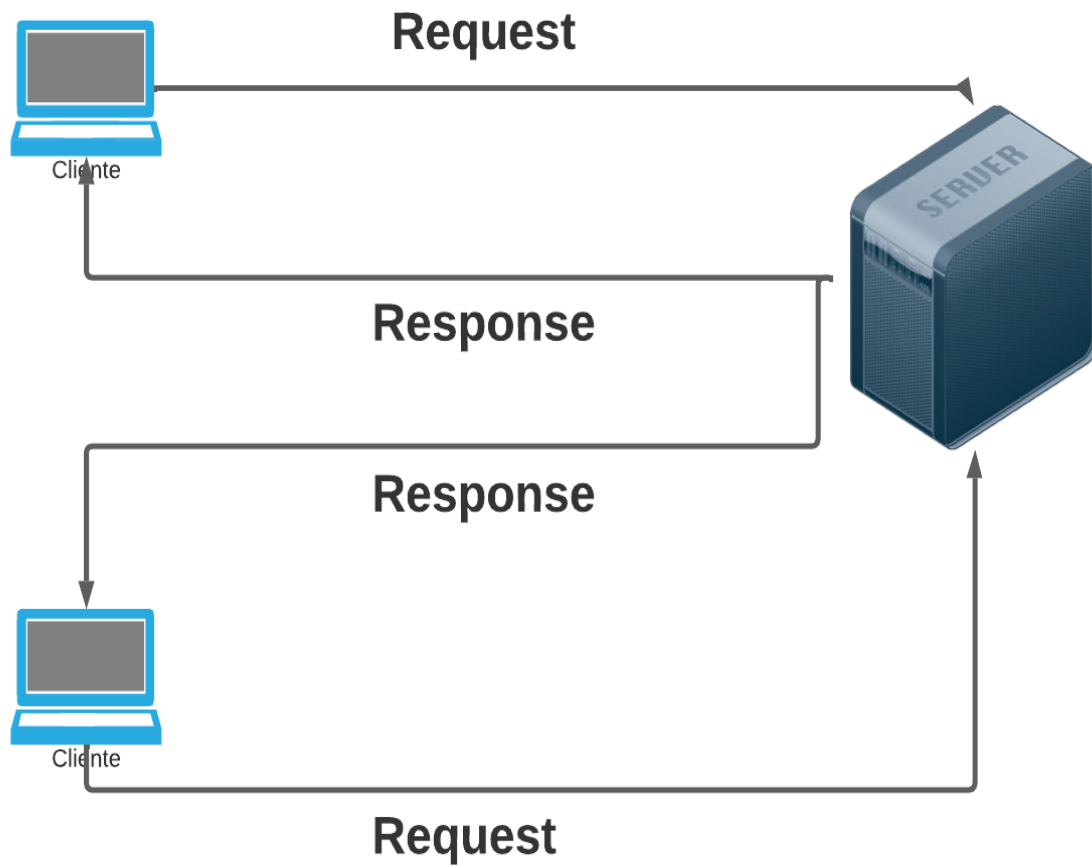


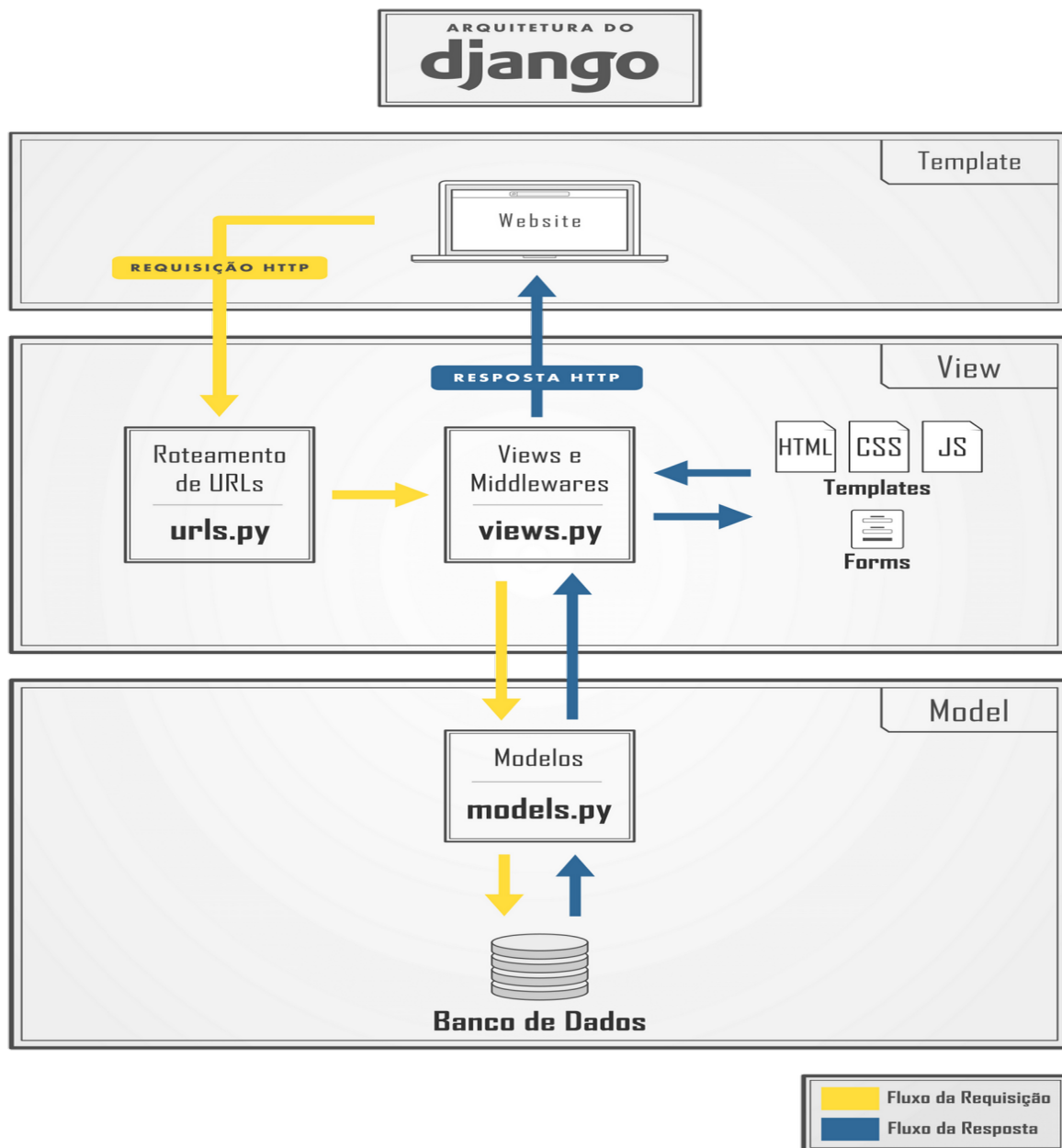
Acesse diretamente pelo Notion:

<https://grizzly-amaranthus-f6a.notion.site/AULA-1-e6dede9603454416bda57aafbd39bbfc?pvs=4>

## Conceitos



Fluxo de dados no Django:



## O projeto

### Configurações iniciais

Primeiro devemos criar o ambiente virtual:

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

Após a criação do venv vamos ativa-lo:

```
#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
```

Agora vamos fazer a instalação do Django e as demais bibliotecas:

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

Vamos criar o nosso projeto Django:

```
django-admin startproject study_async .
```

Rode o servidor para testar:

```
python manage.py runserver
```

Crie o app usuario:

```
python manage.py startapp usuarios
```

INSTALE O APP!

## Cadastro

Crie uma URL para os usuários:

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

No APP usuários crie uma URL para o cadastro:

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

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

Crie a view cadastro:

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

Configure onde o Django deve buscar por arquivos de templates:

```
os.path.join(BASE_DIR, 'templates')
```

Crie o arquivo templates/base.html

```
{% load static %}  
<!doctype html>  
<html lang="en">  
  <head>  
    <meta charset="utf-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1">
```

```

<link
href="<https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css>"
rel="stylesheet">
<title>Study Async</title>
{% block 'cabecalho' %} {% endblock %}
</head>
<body>
  {% block 'conteudo' %}{% endblock %}
  <script
src="<https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.
js>"></script>
</body>
</html>

```

Agora vamos criar o cadastro.html:

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

```
{% block 'conteudo' %}
```

```

  <div class="bg1">
    <div class="container">
      <div class="row">
        <div class="col-md">
          <div class="box-titulo">
            <div class="box-titulo-item">
              <img src="">
            </div>
            <div style="margin-top: auto; margin-bottom: auto;"
class="box-titulo-item">
              <h3 >Study.Async</h3>
            </div>
          </div>
          <hr class="divisor">
          <br>
          <div class="tac">
            <h4 style="font-size: 30px">Sua plataforma online<br>
para te ajudar nos estudos</h4>
          </div>
        </div>
        <div class="col-md">
          <br>
          <br>
          <div class="box-form">
            <div class="box-circle">
              <div class="circle c1"></div>
              <div class="circle c2"></div>
              <div class="circle c3"></div>
            </div>
            <br>
            <h3>Cadastre-se</h3>
            <form action="" method="POST">
              <label>Username</label>
              <input type="text" class="input-form"
name="username" id="">
              <br>
              <br>
              <label>Senha</label>
              <input type="password" class="input-form"
name="senha" id="">
              <br>
              <br>

```



```
    background-image: url('/static/usuarios/img/bg1.png') !important;
    height: 100vh;
    background-size: cover;
}

.titulo-fonte{
    margin-top: 20px;
}

.box-titulo{
    display: flex;
}

.divisor{
    border: 0;
    height: 1px;
    background-image: linear-gradient(to right, transparent, rgb(162, 219, 236),
transparent);
    display: block;
}

.tac{
    text-align: center;
}

.box-form{
    padding: 20px;
    background-color: rgba(217,217,217, .05);
    color: white !important;
}

.box-circle{
    display: flex;
    gap: 5px;
}

.circle{
    width: 16px;
    height: 16px;
    border-radius: 8px;
}

.c1{
    background-color: #3E79EB;
}

.c2{
    background-color: #29BFC9;
}

.c3{
    background-color: #D98562
}

.input-form{
    background-color: #0b1526;
    padding: 10px;
    border: none;
    border: 1px solid rgba(88, 252, 213, .4);
    color: white;
    outline: 0;
}
```

```

        width: 100%;
    }

    .btn-cadastro{
        padding: 10px;
        background-color: #58fcd5;
        border: none;
        border: 1px solid #58fcd5;
        font-size: 18px;
        font-weight: bold;
        padding-left: 25px;
        padding-right: 25px;
    }

```

Importe o arquivo css em cadastro.html:

```

{% load static %}

{% block 'cabecalho' %}
    <link href="{% static 'usuarios/css/cadastro.css' %}" rel="stylesheet">
{% endblock %}

```

Faça o download dos assets no link abaixo:

<https://drive.google.com/drive/folders/1eoUuQCK4Plj1jnEARavxNCyOMkLS55qZ?usp=sharing>

Adicione o bg1.png e o logo.png na pasta /templates/static/usuarios/img

Adicione a logo em cadastro.html:

```



```

Execute as migrações:

```

python manage.py makemigrations
python manage.py migrate

```

Prepare o form para enviar os dados para a View:

```

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

```

Na view cadastro:

```

def cadastro(request):
    if request.method == 'GET':
        return render(request, 'cadastro.html')
    else:
        username = request.POST.get('username')
        senha = request.POST.get('senha')
        confirmar_senha = request.POST.get('confirmar_senha')

        if not senha == confirmar_senha:
            return redirect('/usuarios/cadastro')

        user = User.objects.filter(username=username)

        if user.exists():
            return redirect('/usuarios/cadastro')

        try:
            user = User.objects.create_user(

```

```

        username=username,
        password=confirmar_senha,
    )

    return redirect('/usuarios/login')
except:

    return redirect('/usuarios/cadastro')

```

Configure as mensagens:

```
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 em pontos específicos de sua View:

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

        if not senha == confirmar_senha:
            messages.add_message(
                request, constants.ERROR, 'As senhas não coincidem'
            )
            return redirect('/usuarios/cadastro')

        user = User.objects.filter(username=username)

        if user.exists():
            messages.add_message(
                request,
                constants.ERROR,
                'Já existe um usuário com o mesmo username',
            )
            return redirect('/usuarios/cadastro')

        try:
            user = User.objects.create_user(
                username=username,
                password=confirmar_senha,
            )
            messages.add_message(
                request, constants.ERROR, 'Usuário cadastrado com sucesso.'
            )
            return redirect('/usuarios/login')
        except:
            messages.add_message(
                request, constants.ERROR, 'Erro interno do sistema'
            )
            return redirect('/usuarios/cadastro')

```

Exiba as mensagens:



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

## Login

Adicione a URL para login:

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

Crie a view logar:

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

Crie o login.html

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

{% block 'cabecalho' %}
    <link href="{% static 'usuarios/css/cadastro.css' %}" rel="stylesheet">
{% endblock %}

{% block 'conteudo' %}

    <div class="bg1">
        <div class="container">
            <div class="row">
                <div class="col-md">
                    <div class="box-titulo">
                        <div class="box-titulo-item">
                            
                        </div>
                        <div style="margin-top: auto; margin-bottom: auto;"
class="box-titulo-item">
                            <h3 >Study.Async</h3>
                        </div>
                    </div>
                    <hr class="divisor">
                    <br>
                    <div class="tac">
                        <h4 style="font-size: 30px">Sua plataforma online<br>
para te ajudar nos estudos</h4>
                    </div>

                </div>
                <div class="col-md">
                    <br>
                    <br>
                    <div class="box-form">
                        <div class="box-circle">
                            <div class="circle c1"></div>
                            <div class="circle c2"></div>
                            <div class="circle c3"></div>
                        </div>
                    <br>
```

```

<h3>Logar</h3>
{% if messages %}
    {% for message in messages %}
        <section class="alert {{message.tags}}">
            {{message}}
        </section>
    {% endfor %}
{% endif %}
<form action="" method="">
    <label>Username</label>
    <input type="text" class="input-form"
name="username" id="">
    <br>
    <br>
    <label>Senha</label>
    <input type="password" class="input-form"
name="senha" id="">
    <br>
    <br>
    <input type="submit" value="Enviar" class="btn-
cadastro">
    <br>
    <br>
    <br>
    <br>
</form>
</div>
</div>
</div>
</div>
</div>
{% endblock 'conteudo' %}

```

Envie os dados do form para a view login:

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

Na view login faça o login do usuário:

```

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

        user = auth.authenticate(request, username=username, password=senha)
        if user:
            auth.login(request, user)
            messages.add_message(request, constants.SUCCESS, 'Logado!')
            return redirect('/flashcard/novo_flashcard/')
        else:
            messages.add_message(
                request, constants.ERROR, 'Username ou senha inválidos'
            )
            return redirect('/usuarios/login')

```

## Logout

Crie uma URL para o Logout:

```
path('logout/', views.logout, name='logout'),
```

Crie a View Logout:

```
def logout(request):  
    auth.logout(request)  
    return redirect('/usuarios/login')
```

## Novo Flashcard

Crie um novo app:

```
python3 manage.py startapp flashcard
```

Instale o APP!

Crie as models:

```
class Categoria(models.Model):  
    nome = models.CharField(max_length=20)  
  
    def __str__(self):  
        return self.nome  
  
class Flashcard(models.Model):  
    DIFICULDADE_CHOICES = (('D', 'Difícil'), ('M', 'Médio'), ('F', 'Fácil'))  
    user = models.ForeignKey(User, on_delete=models.DO_NOTHING)  
    pergunta = models.CharField(max_length=100)  
    resposta = models.TextField()  
    categoria = models.ForeignKey(Categoria, on_delete=models.DO_NOTHING)  
    dificuldade = models.CharField(max_length=1, choices=DIFICULDADE_CHOICES)  
  
    def __str__(self):  
        return self.pergunta
```

Execute as migrações:

```
python manage.py makemigrations  
python manage.py migrate
```

Cadastre no admin:

```
from .models import Categoria, Flashcard  
  
admin.site.register(Categoria)  
admin.site.register(Flashcard)
```

Crie um superusuário