



## Sobre Este Curso

### **Público Alvo**

Programadores Python que desejam aprender a programar utilizando o Framework Django.

### **Pré-Requisitos**

Conhecimentos em Lógica de Programação e Linguagem Python.



## Índice

<b>SOBRE ESTE CURSO .....</b>	<b>I</b>
PÚBLICO ALVO .....	I
PRÉ-REQUISITOS .....	I
<b>ÍNDICE.....</b>	<b>I</b>
<b>COPYRIGHT .....</b>	<b>III</b>
EQUIPE.....	III
HISTÓRICO DAS EDIÇÕES .....	III
<b>CAPÍTULO 01 – HTML .....</b>	<b>4</b>
O QUE É O HTML .....	4
O QUE É HTML 5.....	4
COMO FUNCIONA O HTML 5 .....	4
ELEMENTOS DE SCRIPT.....	7
CRIANDO FORMULÁRIO .....	7
<b>CAPÍTULO 02 – CSS.....</b>	<b>17</b>
O QUE É CSS .....	17
ESTILIZANDO NOSSA PÁGINA.....	18
<b>CAPÍTULO 03 – JAVASCRIPT .....</b>	<b>22</b>
O QUE É .....	22
CRIANDO A INTERAÇÃO COM O SITE .....	24
<b>CAPÍTULO 04 – PREPARANDO AMBIENTE PARA INICIAR COM DJANGO .....</b>	<b>26</b>
<b>CAPÍTULO 05 – CONFIGURANDO BANCO DE DADOS PELO DJANGO.....</b>	<b>28</b>
<b>CAPÍTULO 06 – CRIANDO USUÁRIO DJANGO .....</b>	<b>29</b>
<b>CAPÍTULO 07 – CRIANDO APLICAÇÃO DJANGO .....</b>	<b>30</b>
<b>CAPÍTULO 08 – CRIANDO MODELO DE DADOS COM DJANGO.....</b>	<b>31</b>
<b>CAPÍTULO 09 – CRIANDO UM FORMULÁRIO .....</b>	<b>33</b>
<b>CAPÍTULO 10 – CRIANDO AS VIEWS .....</b>	<b>34</b>
<b>CAPÍTULO 11 – CRIANDO ROTAS .....</b>	<b>36</b>
<b>CAPÍTULO 12 – CRIANDO LAYOUT.....</b>	<b>38</b>
<b>CAPÍTULO 13 – PROJETO PRONTO .....</b>	<b>40</b>
<b>CAPÍTULO 14 – HORA DE TREINAR .....</b>	<b>41</b>
EXERCÍCIOS .....	41

	Anotações



Anotações	

## Copyright

As informações contidas neste material se referem ao curso de **Python com Django** e estão sujeitas as alterações sem comunicação prévia, não representando um compromisso por parte do autor em atualização automática de futuras versões.

A **Apex** não será responsável por quaisquer erros ou por danos acidentais ou consequenciais relacionados com o fornecimento, desempenho, ou uso desta apostila ou os exemplos contidos aqui.

Os exemplos de empresas, organizações, produtos, nomes de domínio, endereços de e-mail, logotipos, pessoas, lugares e eventos aqui representados são fictícios. Nenhuma associação a empresas, organizações, produtos, nomes de domínio, endereços de e-mail, logotipos, pessoas, lugares ou eventos reais é intencional ou deve ser inferida.

A reprodução, adaptação, ou tradução deste manual mesmo que parcial, para qualquer finalidade é proibida sem autorização prévia por escrito da **Apex**, exceto as permitidas sob as leis de direito autoral.

## Equipe

### Conteúdos

- Gustavo Rosauro

### Diagramação

- Elizabeth Cruz Pereira

### Revisão

- Fernanda Pereira

## Histórico das Edições

Edição	Idioma	Edição
1ª	Português	Março de 2020

© Copyright 2020 **Apex**. Desenvolvido por Gustavo Rosauro e licenciado para Apex Ensino

	Anotações



## Capítulo 01 – HTML



### O que é o HTML

HTML é a sigla para **Hypertext Markup Language** (Linguagem de marcação de hipertexto). Ao contrário do que muitos pensam, HTML não é uma linguagem de programação, mas sim um conjunto de tags para marcação de texto. Esse conjunto de tags é interpretado pelos navegadores para a visualização de um documento através da web.

### O que é HTML 5

HTML 5 foi o nome dado à quinta versão da especificação do html, mas também é um termo utilizado entre as equipes de desenvolvimento para projetos que utilizam as tecnologias HTML CSS3 e Javascript. Isso se deve porque juntamente com a especificação do HTML 5 foram incluídas novas funcionalidades para manipulação de documentos html através do javascript.

### Como funciona o HTML 5

A imagem abaixo apresenta a marcação básica de um documento HTML.

Anotações	

```

1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <title>Olá HTML</title>
5          <meta charset="UTF-8">
6      </head>
7      <body>
8          Este é um documento html
9      </body>
10 </html>

```

### **<!DOCTYPE html>**

Esta marcação indica que o documento a seguir segue a especificação html da versão 5, ao contrário das versões anteriores, o doctype para o html 5 é pequeno.

### **<html>**

Indica a abertura do documento html. Todo o conteúdo do documento deve ficar entre a tag de abertura <html> e a tag de fechamento </html>. A tag html possui o atributo lang que é utilizado para indicar qual o idioma utilizado no documento.

### **<head>**

Indica a abertura do cabeçalho do documento. Normalmente o conteúdo adicionado dentro do cabeçalho é utilizado para fins de passar informações ao browser sobre o conteúdo que virá no documento. As informações do head não são visíveis diretamente no documento, mas são utilizadas pelo browser e por robôs para a indexação do documento. O head deve ser encerrado com sua tag de fechamento </head>.

### **<title></title>**

Utilizada para a demarcação do título do documento. O conteúdo colocado entre a tag title não é renderizado no corpo do documento, mas é utilizado pelo navegador para informar o título da página em sua aba superior.

### **<meta charset="UTF-8">**

	Anotações



Tag de meta dados utilizada para definir qual o tipo de encode foi utilizado pelo documento no momento de sua criação.

### **<body>**

Indica o início do corpo do documento. Todo o conteúdo que será apresentado ao usuário deve estar contido entre as tags <body> e </body>

### **</html>**

Indica o fim do documento.

## ***Tipos de Tags HTML***

A especificação HTML nos apresenta várias tags/elementos que são utilizados com os mais variados propósitos, entretanto, podemos agrupar essas tags/elementos em alguns grupos distintos:

### ***Elementos Containers***

São tags/elementos utilizados para agrupar outros elementos de forma lógica, estes elementos normalmente delimitam partes do documento e tem como característica a quebra de linha antes e após a sua demarcação.

#### **Exemplos:**

<p></p>

<div></div>

### ***Elementos in-line***

São tags/elementos utilizados normalmente para a formatação do texto, aplicando ao documento certo grau de estilização. São denominados in-line por não apresentarem quebra de linha no documento, não alterando seu direcionamento.

#### **Exemplos:**

<strong></strong>

<span></span>

### ***Elementos de imagem e multimídia***

São elementos utilizados para a inclusão de imagem, som e vídeo dentro de um documento html.

#### **Exemplos:**



Anotações	



```
<audio src="http://developer.mozilla.org/@api/deki/files/2926/=AudioTest_(1).ogg" autoplay>
```

O seu navegador não suporta o elemento `<code>audio</code>`

```
</audio>
```

### Elementos de Script

São utilizados para a inclusão de scripts ao documento para posterior processamento do navegador.

#### Exemplos:

```
<script></script>
```

```
<canvas></canvas>
```

### Elementos de Meta Dados

São utilizados para adicionarem informações referente ao documento e são utilizados normalmente pelo navegador e por robôs de indexação de páginas.

#### Exemplo:

```
<meta charset="UTF-8">
```

## Criando Formulário

### Elemento Form

A tag **<form>** é utilizada para demarcar o conteúdo de um formulário. Este elemento possui dois atributos importantes que são:

- action: indica qual é a ação a ser processada no servidor.
- method: indica a forma de envio dos dados. Os valores válidos para estes atributos são "get" ou "post".

#### Exemplo:

```
<!DOCTYPE html>
<html lang="pt">
<head>
<meta charset="UTF-8">
```

	Anotações



```
<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag form</title>

</head>

<body>

<form action="salvar" method="post">

</form>

</body>

</html>
```

### Elemento Fieldset

A tag **<fieldset>** é um elemento utilizado para demarcar um agrupamento de dados no formulário.

### Elemento Legend

A tag **<legend>** é o elemento utilizado para demarcar o título de um fieldset.

### Elemento Label

A tag **<label>** é um elemento utilizado para demarcar o marcador de um campo do formulário.

### Exemplo:

```
<!DOCTYPE html>

<html lang="pt">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag fieldset</title>

</head>

<body>
```

Anotações	

```
<form action="salvar" method="post">

<fieldset>

<legend>Dados Pessoais</legend>

<label>Nome</label>

<input type="text">

</fieldset>

</form>

</body>

</html>
```

### Elemento input

A tag **<input>** é utilizado para apresentar um campo de entrada de dados de um formulário. A tag input é extremamente versátil podendo renderizar vários tipos de entrada de dados apenas alterando o valor do atributo type. As renderizações mais comuns são:

#### Input type:text

Apresenta um campo para entrada de dados do tipo texto.

#### Input type:email

Apresenta um campo para entrada de dados do tipo email.

#### Input type:checkbox

Apresenta um campo para entrada de dados do tipo checkbox (caixa de seleção).

#### Input type:radio

Apresenta um campo para entrada de dados do tipo radio(caixa de seleção).

#### Input type:number

Apresenta um campo para entrada de dados do tipo numérico.

#### Input type:password

Apresenta um campo para entrada de dados para senhas.

	Anotações



### **Input type:file**

Apresenta um campo para seleção de arquivos para envio ao servidor.

### **Input type:reset**

Apresenta um botão para retornar o conteúdo dos campos do formulário ao seus valores padrão.

### **Input type:submit**

Apresenta um botão para enviar os dados formulário.

### **Input type:tel**

Apresenta um campo para informar um número de telefone. Quebras de linha são removidas automaticamente do texto informado.

### **Input type:button**

Apresenta um botão sem comportamento padrão.

### **Input type:hidden**

Apresenta um controle não visível utilizado para enviar informações para o servidor.

Além do atributo type, o elemento input apresenta os seguintes atributos também muito utilizados:

**name:** define um nome para o campo. Este nome é utilizado pelo servidor para ler o valor do campo no formulário.

**value:** utilizado para definir um valor para o campo

**placeholder:** utilizado para apresentar um exemplo de texto a ser inserido no elemento.

**required:** utilizado para indicar se o valor do campo é obrigatório

**readonly:** utilizado para indicar se o valor do campo é somente para leitura, sem que o usuário possa editar esse valor.

Anotações	

### Exemplo:

```
<!DOCTYPE html>

<html lang="pt">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag input</title>

</head>

<body>

<form action="salvar" method="post">

<fieldset>

<legend>Dados Pessoais</legend>

<input type="hidden" name="codigo"><br>

<label>Nome</label><br>

<input type="text"><br>

<label>Email</label><br>

<input type="email"><br>

<label>Telefone</label><br>

<input type="tel"><br>

<label>Idade</label><br>

<input type="number"><br>

<label>Senha</label><br>

<input type="password" required><br>

</fieldset>
```

	Anotações



```
<fieldset>

<legend>Benefícios</legend>

<label> <input type="checkbox">Unimed</label><br>

<label><input type="checkbox">Uniodonto</label><br>

</fieldset>

<fieldset>

<legend>Alimentação</legend>

<label> <input type="radio" name="alimentacao">Cartão Refeição</label><br>

<label><input type="radio" name="alimentacao">Cartão Alimentação</label><br>

</fieldset>

</form>

</body>

</html>
```

### Elemento button

A tag **<button>** é utilizada para criar um botão clicável dentro do formulário.

#### Exemplo:

```
<!DOCTYPE html>

<html lang="pt">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag button</title>

</head>

<body>
```

Anotações	

```
<form action="salvar" method="post">

<fieldset>

<legend>Botoes</legend>

<button type="button" onclick="alert('Bem Vindo!')">Click</button>

<button type="submit" >Enviar</button>

</fieldset>

</form>

</body>

</html>
```

### Elemento select

A tag **<select>** é utilizada para criar um elemento contendo um menu de opções. Este elemento normalmente é utilizado em conjunto com as tags **<option>** e **<optgroup>**.

#### Exemplo:

```
<!DOCTYPE html>

<html lang="pt">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag input</title>

</head>

<body>

<form action="salvar" method="post">

<fieldset>

<legend>Cidades</legend>

<select>
```

	Anotações



```
<optgroup label="Santa Catarina">
<option value="Blumenau" label="Blumenau"></option>
<option value="Florianópolis" label="Florianópolis"></option>
<option value="Criciúma" label="Criciúma"></option>
<option value="Lages" label="Lages"></option>
</optgroup>
<optgroup label="Rio Grande do Sul">
<option value="Porto Alegre" label="Porto Alegre"></option>
<option value="Canoas" label="Canoas"></option>
<option value="Guaíba" label="Guaíba"></option>
<option value="Santana do Livramento" label="Santana do Livramento"></option>
</optgroup>
</select>
</fieldset>
</form>
</body>
</html>
```

### Elemento textarea

A tag **<textarea>** é utilizada para apresentar uma caixa de texto com tamanho grande.

#### Exemplo:

```
<!DOCTYPE html>
<html lang="pt">
<head>
<meta charset="UTF-8">
```

Anotações	



```
<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag textarea</title>

</head>

<body>

<form action="salvar" method="post">

<fieldset>

<legend>Informações</legend>

<label >Comentários</label><br>

<textarea cols="40" rows="10"></textarea>

</fieldset>

</form>

</body>

</html>
```

### Elemento progress

A tag **<progress>** é utilizada para apresentar uma barra de progresso.

#### Exemplo:

```
<!DOCTYPE html>

<html lang="pt">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Exemplo tag progress</title>

</head>

<body>
```

	Anotações



```
<form action="salvar" method="post">
<fieldset>
<legend>Informações</legend>
<label >andamento</label><br>
<progress value="70" max="100">70%</progress>
</fieldset>
</form>
</body>
</html>
```

**Para informações mais aprofundadas sobre o HTML 5, consulte apostila disponível em sua área do aluno.**

Anotações	

## Capítulo 02 – CSS



### O que é CSS

Folhas de estilo em cascata ou no inglês “**Cascading Style sheet**” (CSS), é uma linguagem de marcação utilizada para a estilização de documentos html, e tem como objetivo separar a estilização de apresentação do documento da informação contida no próprio documento.

Com o uso do CSS podemos adicionar fontes, cores, margens, bordas entre outros, a elementos contidos dentro de documentos web.

### Regra CSS

Regra CSS é a unidade básica utilizada para a estilização de um documento através do CSS. Uma regra CSS é composta por um seletor e uma ou mais declarações de propriedades/valores.

#### Exemplo:

```
seletor {  
  propriedade:valor;  
}
```

#### Onde:

**Seletor:** É o alvo da regra css, o componente que se deseja estilizar;

**Propriedade:** É o atributo/característica do componente que desejamos configurar;

**Valor:** É a qualificação do atributo;

#### Exemplo:

```
h1{  
  color: red;  
  Font-family: "Times New Roman";
```

	Anotações



```
}
```

### Comentários de regras CSS

Em folhas de estilo em cascata podemos adicionar comentários para melhor organização e manutenção.

Para adicionarmos um comentário devemos utilizar os caracteres `/*` para iniciar o bloco de comentários e em seguida os caracteres `*/` para fechar o bloco de comentários.

#### Exemplo:

```
/* Este é um comentário válido em uma folha de estilos */
```

## Estilizando nossa Página

Existem 4 formas básicas de vincularmos estilos a um documento html:

### Estilos inline

Estilos inline são vinculados através da propriedade `style` pertencente a todos os componentes html. Os valores atribuídos à propriedade `style` serão utilizados para a estilização do componente.

#### Exemplo:

```
/* documento html */  
<h2 style="color:blue; background-color:#ccc"> Meu título</h2>
```

### Estilos incorporados

Outra forma de adicionarmos estilos a um documento html é configurando-os dentro da tag `<style></style>`. Esta tag deve ser definida dentro do componente `head`.

#### Exemplo:

```
/* documento html */  
<!DOCTYPE html>  
<html>  
<head>  
  <style>
```

Anotações	

```
body{
  margin: 0 0 0 0;
}
h2{
  color:blue;
  background-color:#ccc;
}

.container{
  margin-left: 10px;
}
</style>
</head>
<body>
</body>
</html>
```

### Folhas de estilo ligadas

Folhas de estilo ligadas são criadas dentro de arquivos com a extensão '.css' e posteriormente ligadas ao documento onde serão utilizadas.

#### Exemplo:

```
/* documento meu-estilo.css */
body{
  margin: 0 0 0 0;
}
h2{
  color:blue;
  background-color:#ccc;
}

.container{
```

	Anotações



```
margin-left: 10px;
}

/* documento html */
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="meu-estilo.css">
</head>
<body>
</body>
</html>
```

### Folhas de estilo importadas

Folhas de estilo importadas são folhas de estilo criadas dentro de um arquivo com a extensão “.css” e posteriormente importados para o documento através da propriedade import dentro da tag style.

#### Exemplo:

```
/* documento meu-estilo.css */
body{
margin: 0 0 0 0;
}
h2{
color:blue;
background-color:#ccc;
}

.container{
margin-left: 10px;
}
```

Anotações	

```
/* documento html */
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <style>
```

```
    @import url('meu-estilo.css');
```

```
  </style>
```

```
</head>
```

```
<body>
```

```
</body>
```

```
</html>
```

**Para informações mais aprofundadas sobre o CSS, consulte apostila disponível em sua área do aluno.**

	Anotações



## Capítulo 03 – Javascript



### O que é

Javascript é uma linguagem interpretada baseada em Objetos e funções com tipagem dinâmica. JavaScript foi originalmente desenvolvido por [Brendan Eich](#), da [Netscape](#) sob o nome de *Mocha*, posteriormente teve seu nome mudado para *LiveScript* e por fim *JavaScript*.

LiveScript foi o nome oficial da linguagem quando foi lançada pela primeira vez na versão beta do navegador Netscape 2.0 em setembro de 1995, mas teve seu nome mudado em um anúncio conjunto com a [Sun Microsystems](#) em dezembro de 1995 quando foi implementado no navegador Netscape versão 2.0B3. (<https://pt.wikipedia.org/wiki/JavaScript>)

### JavaScript e a especificação ECMAScript

O JavaScript é padronizado pela [Ecma International](#) — a associação Europeia para a padronização de sistemas de comunicação e informação (antigamente ECMA era um acrônimo para European Computer Manufacturers Association) para entregar uma linguagem de programação padronizada, internacional baseada em Javascript.

Esta versão padronizada de Javascript, chamada ECMAScript, comporta-se da mesma forma em todas as aplicações que suportam o padrão. As empresas podem usar a linguagem de padrão aberto para desenvolver a sua implementação de Javascript. O padrão ECMAScript é documentado na especificação ECMA-262.

### Como incorporar Javascript nas páginas html

Em Podemos incorporar javascript a uma página html basicamente de duas formas:

- Criar o Script diretamente na página html.

**Exemplo:**

Anotações	



```

1  <!doctype html>
2  <html>
3  <head></head>
4  <body>
5  <script>
6  document.write("<h1>Bem Vindo ao Javascript</h1>");
7  </script>
8  </body>
9  </html>

```

- Criar um arquivo de script separado da página e referenciá-lo posteriormente na página em que desejamos utilizá-lo. Primeiro precisamos criar um arquivo com a extensão .js.

#### Exemplo:

```

1  //app.js
2  document.write("<h1>Bem Vindo ao Javascript</h1>");

```

Em seguida importamos o arquivo javascript informando sua localização na propriedade src (source) da tag javascript.

```

1  <!doctype html>
2  <html>
3  <head></head>
4  <body>
5  <script src="app.js"></script>
6  </body>
7  </html>

```

	Anotações



## Criando a interação com o Site

### *O que são eventos*

Eventos são funções executadas pelo browser em decorrência de ações realizadas pelo usuário ou pelo próprio browser.

### *Principais eventos*

Click - Disparado quando o mouse é pressionado sobre um elemento

Keydown - Disparado quando a tecla é pressionada para baixo.

KeyUp - Disparado quando a tecla é liberada.

KeyPress - Disparado após a tecla ser pressionada e liberada.

Focus - Disparado quando um elemento recebe o foco.

Blur - Disparado quando um elemento perde o foco.

MouseOver - Disparado quando o mouse está sobre o elemento

Input - Disparado quando um input, textarea ou select é alterado.

Load - Disparado quando o elemento é carregado.

Anotações	

### *Como se registrar para um evento*

Para que possamos ser notificados sobre um determinado evento, devemos nos registrar no elemento passando uma função de notificação. Esse registro é feito através da função `addEventListener`.

#### **Exemplo:**

```
let listener= document.querySelector('button').addEventListener(
    'click',function(evt){
        Console.log('Click no botão')
    })
```

### *Como deixar de ouvir um evento*

Para deixar de ouvir um evento, basta chamar a função `removeEventListener`, passando como parâmetros o nome do evento e o listener que se ligava ao evento.

#### **Exemplo:**

```
document.querySelector('button').removeEventListener('click',listener);
```

**Para informações mais aprofundadas sobre o Javascript, consulte apostila disponível em sua área do aluno.**

	Anotações



## Capítulo 04 – Preparando ambiente para iniciar com Django



Agora vamos começar a utilizar o Framework Django.

Vamos criar um novo venv, local onde armazena os packages do app.

Executando o seguinte comando `py -m venv env` pelo prompt de comando, ir na pasta scripts e escolher a opção `activate` para poder configurar o env.

Conforme demonstra na imagem abaixo.

```
Pasta de C:\Users\Acer\PycharmProjects\DjangoMySQL\env\Scripts
05/03/2020 19:10 <DIR> .
05/03/2020 19:10 <DIR> ..
05/03/2020 19:10 2.303 activate
```

No cmd, voltamos para o nível da pasta do projeto e damos sequência.

Para iniciar o django, instale com o seguinte comando: `Pip Install django`.

Anotações	

```
Terminal: Local x +
Microsoft Windows [versão 10.0.18362.657]
(c) 2019 Microsoft Corporation. Todos os direitos reservados.

(venv) C:\Users\Acer\PycharmProjects\DjangoMySql>pip install django
Collecting django
  Downloading https://files.pythonhosted.org/packages/12/68/8c125da33aaf0942add5095a7a2a8e064b3812d598e9fb5aca9957872d71/Django-3.0.4-py3-none-any.whl (7.5MB)
    87% |#####| 6.5MB 3.1MB/s eta 0:00:01
```

Após instalado o django, instalar o pacote para comunicação do app com o Banco de Dados mysql utilizando o seguinte comando `pip install mysqlclient`

```
(venv) C:\Users\Acer\PycharmProjects\DjangoMySql>pip install mysql-connector-python
```

Após instalados os pacotes do django e do mysql podemos criar o nosso projeto com o django utilizando o seguinte comando: `django-admin startproject <nome do projeto>`

Conforme imagem abaixo:

```
Requirement already satisfied: setuptools in c:\users\acer\pycharmprojects\djangomysql\venv\lib\site-packages (4.4.0)
Installing collected packages: dnspython, six, protobuf, mysql-connector-python
Successfully installed dnspython-1.16.0 mysql-connector-python-8.0.19 protobuf-3.6.1 six-1.14.0

(venv) C:\Users\Acer\PycharmProjects\DjangoMySql>django-admin startproject DjangoMySql

(venv) C:\Users\Acer\PycharmProjects\DjangoMySql>
```

	Anotações



## Capítulo 05 – Configurando Banco de Dados pelo Django

Com os pacotes instalados, vá no documento settings.py e configure o banco de dados. Encontre uma variável com o nome databases, lá coloque as informações para conectar com o banco, conforme mostra imagem abaixo.

```
# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'apexpython',
        'USER': 'Root',
        'PASSWORD': '',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators
```

Configurada a comunicação, você deverá mandar ele estabelecer uma conexão com a base, utilizando o seguinte comando: python manage.py migrate

```
(env) C:\Users\Acer\PycharmProjects\DjangoMySQL\DjangoMySQL>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
```

Anotações	

## Capítulo 06 – Criando Usuário Django

Para criar o usuário django, utilize o seguinte comando: `python manage.py createsuperuser`.

Após executar esse comando, ele irá solicitar um nome de usuário e senha.

Conforme demonstra figura abaixo:

```
(env) C:\Users\Acer\PycharmProjects\DjangoMySql\DjangoMysql>python manage.py createsuperuser
Username (leave blank to use 'acer'): admin
Email address:
Password:
Password (again):
The password is too similar to the username.
★ This password is too short. It must contain at least 8 characters.
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

	Anotações



## Capítulo 07 – Criando Aplicação Django

Para criar uma aplicação Django utiliza-se o seguinte comando: `python manage.py startapp <nome da aplicação>`

```
(env) C:\Users\Acer\PycharmProjects\DjangoMySql\DjangoMysql>python manage.py startapp DjangoCrud  
(env) C:\Users\Acer\PycharmProjects\DjangoMySql\DjangoMysql>
```

Em seguida, ir no arquivo `settings.py` e adicionar a aplicação, procurar pela variável `INSTALLED_APPS` e adicionar a aplicação conforme demonstra a figura abaixo.

```
__init__.py × settings.py × base.py ×  
28     ALLOWED_HOSTS = []  
29  
30  
31     # Application definition  
32  
33     INSTALLED_APPS = [  
34         'django.contrib.admin',  
35         'django.contrib.auth',  
36         'django.contrib.contenttypes',  
37         'django.contrib.sessions',  
38         'django.contrib.messages',  
39         'django.contrib.staticfiles',  
40         'DjangoCrud',  
41     ]  
42
```

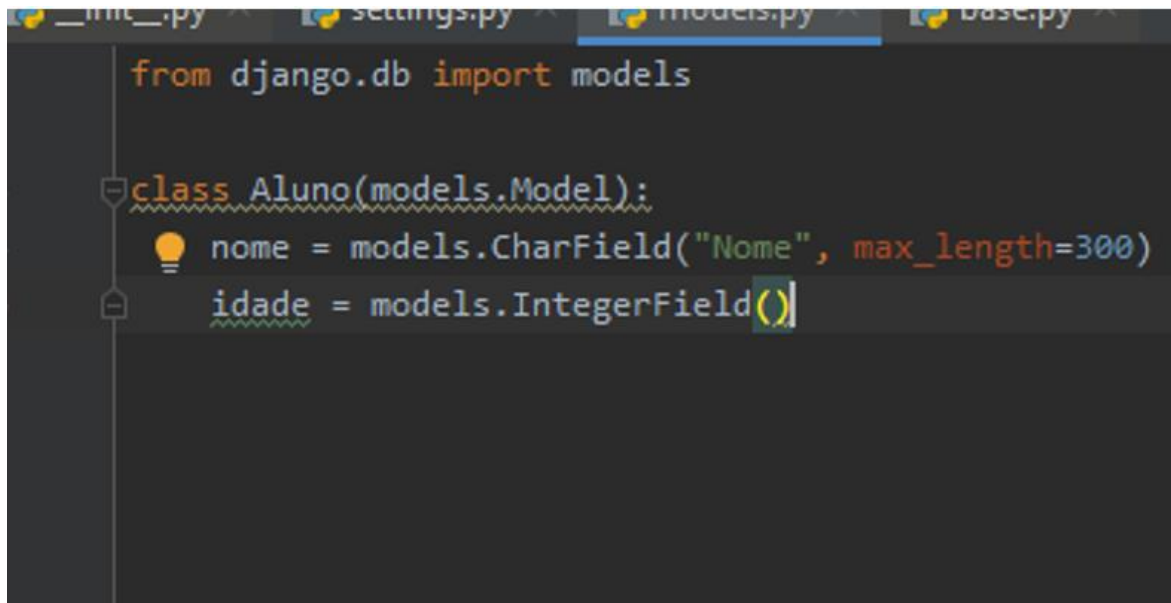
Anotações	



## Capítulo 08 – Criando Modelo de Dados com Django

No projeto django procure pelo arquivo models.py e a estrutura que irá criar será utilizada para formular a tabela no banco de dados. Esse processo chama-se como code-first.

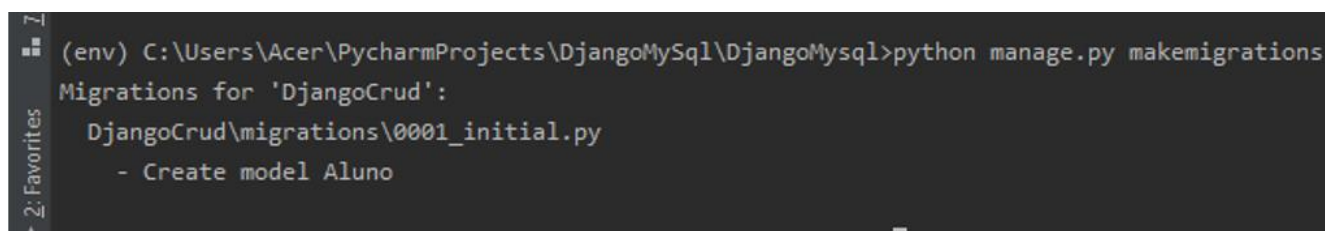
A imagem abaixo demonstra como criar esse modelo de dados.



```
from django.db import models

class Aluno(models.Model):
    nome = models.CharField("Nome", max_length=300)
    idade = models.IntegerField()
```

Após terminado esse processo, execute o seguinte comando: `python manage.py makemigrations` e ele irá criar a tabela no banco de dados mysql.



```
(env) C:\Users\Acer\PycharmProjects\DjangoMySql\DjangoMysql>python manage.py makemigrations
Migrations for 'DjangoCrud':
  DjangoCrud\migrations\0001_initial.py
    - Create model Aluno
```

Ele irá mostrar o que será criado na base de dados. Para que possa efetivar a criação, execute o comando `python manage.py migrate`

	Anotações



```
(env) C:\Users\Acer\PycharmProjects\DjangoMySql\DjangoMysql>python manage.py migrate
Operations to perform:
  Apply all migrations: DjangoCrud, admin, auth, contenttypes, sessions
Running migrations:
  Applying DjangoCrud.0001_initial... OK
(env) C:\Users\Acer\PycharmProjects\DjangoMySql\DjangoMysql>
```

1 SELECT \* FROM djangocrud\_aluno

djangocrud\_aluno (3x0)

	id	nome	idade
--	----	------	-------

Anotações	

## Capítulo 09 – Criando um Formulário

Crie o ModelForm para poder inserir e editar modelos de dados de uma maneira mais simples.

Crie um arquivo com o nome forms.py.

Conforme demonstra a imagem abaixo.

```
from django import forms
from .models import Aluno

class AlunoForm(forms.ModelForm):
    class Meta:
        model = Aluno
        fields = "__all__"
```

Onde model é o meu modelo de dados, e fields são os campos que eu desejo utilizar, quando colocamos o all significa que queremos utilizar todos os campos.

	Anotações



## Capítulo 10 – Criando as Views

No arquivo `views.py` é onde iremos colocar os métodos que irão se comunicar com o layout.

Exemplo: ao clicar em um botão de salvar, esse botão irá chamar um método da nossa view onde será salvo em nosso banco de dados por meio do django.

Para exemplificar, crie dois métodos. Um para trazer informações e outro para que possa inserir um novo registro.

```
5 from .forms import AlunoForm
6 from .models import Aluno
7
8
9 def index(request):
10     aluno = Aluno.objects.all().order_by('nome')
11     lista = {
12         'lista': aluno
13     }
14     return render(request, 'index.html', lista)
15
16 def create(request):
17     if request.method == 'POST':
18         form = AlunoForm(request.POST)
19         if form.is_valid():
20             form.save()
21             return redirect('/aluno')
22     form = AlunoForm()
23     return render(request, 'create.html', {'form': form})
24     pass
25
26
27 def delete(request, id):
28     aluno = Aluno.objects.get(id=id)
29     aluno.delete()
30     return redirect('/aluno')
```

Através do model, pode-se fazer uma consulta para trazer os dados utilizando o `objects.all()` e se quiser ordenar essa consulta por uma coluna da tabela em específico utiliza-se o `.order_by('coluna')`.

Anotações	

Para que possa inserir um novo dado utilizar o método create usando o FormModel que irá salvar as informações na base de dados.

	Anotações



## Capítulo 11 – Criando Rotas

Para configurar a rota onde será chamado os arquivos.html, deve-se ir nas configurações do django e incluir o arquivo urls.py com o include e o caminho que precisa ser acessado.

```
16 from django.contrib import admin
17 from django.urls import path, include
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21     path('aluno', include('DjangoCrud.urls')),
22 ]
23
```

Após essa parte, crie o arquivo urls.py no projeto django e adicione as rotas que foram criadas nas views, conforme demonstra a imagem abaixo.

Anotações	

```

1  from django.urls import path
2
3  from .views import index, create, delete
4
5  urlpatterns = [
6      path('', index, name='index'),
7      path('/create', create, name='create'),
8      path('/delete/<int:id>', delete, name='delete')
9  ]
10

```

O name é o nome que daremos para a nossa página para facilitar a chamada do arquivo precisando somente passar o name.

	Anotações



## Capítulo 12 – Criando Layout

```
ms.py x models.py x views.py x base.html x index.html x create.html x manage.py x DjangoMysql\urls.py x DjangoCrud\urls.py x
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Crud com python Web</title>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
8 </head>
9 <body>
10  {% block content %}
11  {% endblock %}
12  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abt
13  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WL
14  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
15  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
16 </body>
17 </html>
```

Nessa página iremos percorrer uma lista dos usuários que já estão cadastrados na base.

```
<!DOCTYPE html>
<html>
<head>
  <title>Posts</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
  integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
  <style type="text/css">
    <style>
  </style>
</style>
</head>
<body>
  <div class="container-fluid">
    <div class="row">
      <div class="col-md-1 col-xs-1 col-sm-1"></div>

      <div class="col-md-10 col-xs-10 col-sm-10 ">
        <br />
        <h6 style="text-align:center;">
          <font color="red"> Todos os campos são obrigatórios</font>
        </h6>
      </div>
      <div class="col-md-1 col-xs-1 col-sm-1">
    </div>
  </div>
```

Anotações	



```

</div>
<div class="row">
  <div class="col-md-1 col-xs-1 col-sm-1"></div>
  <div class="col-md-10 col-xs-10 col-sm-10">
    <form method='post'>
      {% csrf_token %}
      {{form}}
      <br>
      <button type="submit" class="btn btn-primary">post</button>
    </form>
    <br>
  </div>
  <div class="col-md-1 col-xs-1 col-sm-1"></div>
</div>
</div>
<script>
  let inputs = document.getElementsByTagName('input')
  for(let i = 0; i < inputs.length; i++){
    inputs[i].classList.add('form-control')
  }
</script>
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
  integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
  crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
  integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0wLaUAdn689aCwoqbBJiSnjAK/l8WvCwPm49"
  crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
  integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
  crossorigin="anonymous"></script>
</body>
</html>

```

Essa será nossa página irá inserir um novo registro a cada vez que apertar o botão.

	Anotações



## Capítulo 13 – Projeto Pronto

← → ↻ localhost:8000/aluno

Create

Nome	Idade
Amanda Bollmann Medeiros	23
Cecilia Medeiros Rosauro	0
gustavo	22

Tela que exibe os nossos usuários.

← → ↻ localhost:8000/aluno/create ☆

Todos os campos são obrigatórios

Nome:

Idade:

post

Tela para cadastros dos usuários.

Anotações	

## Capítulo 14 – Hora de Treinar

Agora vamos colocar em prática tudo oq eu vimos até o momento. Abaixo, vocês podem ver uma lista de exercícios, que pedem ações que já foram ensinados em aula.

Fiquem atentos, consultem a apostila sempre que tiverem dúvidas e se precisar podem chamar o instrutor para acompanhá-los.

### Exercícios

- 1- Utilizando django, crie um projeto djangoApex com a estrutura professores sendo:  
Nome com tamanho máximo de 200  
Endereço com tamanho de 500  
E-mail com tamanho máximo de 500
- 2- Após feito essa parte, crie uma tela no html para cadastrar produtos na estrutura que o django criou.
- 3- Na tela de cadastrar crie uma opção para ver os professores já adicionados.
- 4- Crie uma tela para exibir os professores já cadastrados.
- 5- Na mesma tela crie uma opção para remover um professor específico.
- 6- Tem que ter uma opção também para editar o professor já cadastrado.

	Anotações