



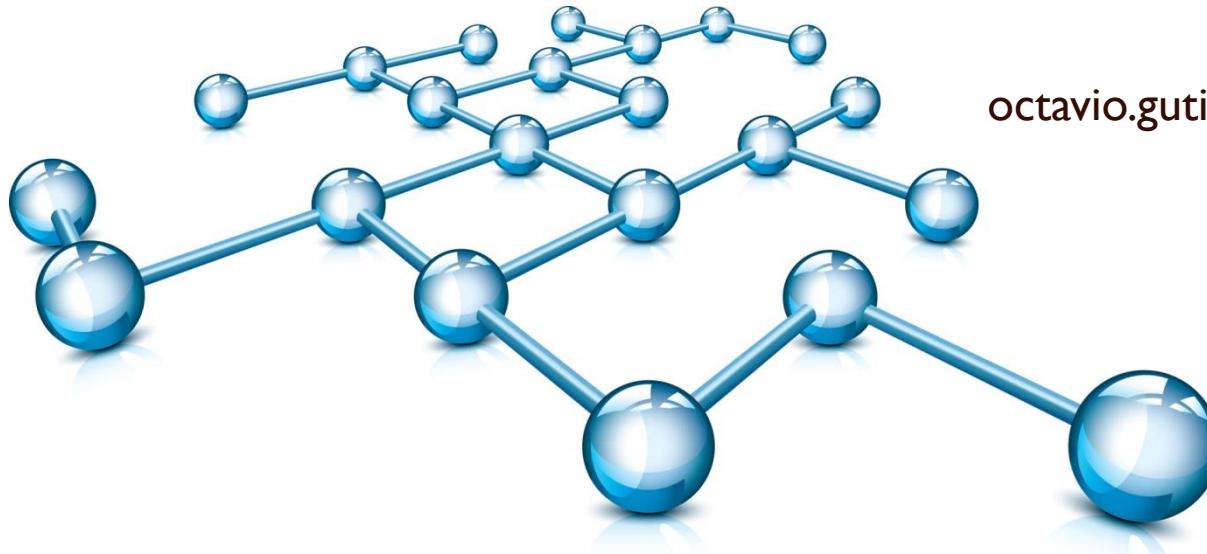
**django**

# Programación Web

Profesor:

Dr. J. Octavio Gutiérrez García

[octavio.gutierrez@itam.mx](mailto:octavio.gutierrez@itam.mx)



# Etiquetas HTML

```
<!DOCTYPE html>
<html>
    <head>

        <meta charset="UTF-8">

        <title>
            Mi primer página web
        </title>

    </head>

    <body>
        ...
    </body>

</html>
```



# Etiquetas HTML

<p> </p>

<h1></h1> ... <h6></h6>

<span> </span>

<div> </div>

<br> <hr>



# Etiquetas HTML

```
<table>
  <tr>
    <th>Nombre</th>
    <th>Edad</th>
  </tr>
  <tr>
    <td>Jose</td>
    <td>55</td>
  <tr>
    <td>Luis</td>
    <td>48</td>
  <tr>
</table>
```



A diagram illustrating the structure of the provided HTML code. It shows a 3x3 grid of cells. The columns are labeled "Column 1", "Column 2", and "Column 3". The rows are labeled "Row 1 Cell 1", "Row 1 Cell 2", "Row 1 Cell 3", "Row 2 Cell 1", "Row 2 Cell 2", "Row 2 Cell 3", and "Row 3 Cell 1". The first row has two cells, the second row has two cells, and the third row has one cell spanning all three columns.

Column 1	Column 2	Column 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
Row 2 Cell 1		Row 2 Cell 3
Row 3 Cell 1		

# Etiquetas HTML

```
<ol>
```

```
  <li>Elemento A</li>
  <li>Elemento B</li>
  <li>Elemento C</li>
```

```
</ol>
```

```
<ul>
```

```
  <li>Elemento A</li>
  <li>Elemento B</li>
  <li>Elemento C</li>
```

```
</ul>
```

## Unordered List

- The first item
- The second item
- The third item
- The fourth item

## Ordered List

1. The first item
2. The second item
3. The third item
4. The fourth item



# Etiquetas HTML

```
<a href="https://www.google.com" target="_blank">Vinculo</a>
```

```
<a href="https://www.google.com" target="_self">Vinculo</a>
```



```

```



# Etiquetas HTML

```
<form action="otra.html" method="get">
```

...

```
</form>
```

## GET IN TOUCH

Tell us your name \*

Enter your email \*

Your Website

Message

SUBMIT



# Etiquetas HTML

```
<label for="nombre">Ingresa tu nombre:</label><br>
<input type="text" id="nombre" name="nombre"><br>
```

```
<p>Selecciona un tipo</p>
<input type="radio" id="tipo" name="tipo" value="tipo">
<label for="tipo"> Tipo 1 </label>
```

```
<label for="tipo">Selecciona un tipo</label>
<select id="tipo" name="tipo" >
    <option value="opcion1" selected>Opcion 2</option>
</select>
```

```
<input type="checkbox" id="opcion" name="opcion"><br>
<label for="opcion">Esta una opción </label>
```

```
<label for="texto_largo">Escribe algo:</label>
<textarea name="texto_largo" id = "texto_largo" rows="10" cols="30">
    Este es un mensaje largo
</textarea><br>
```

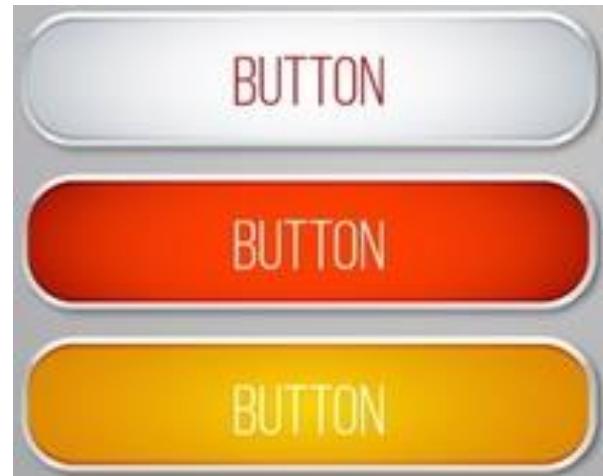


# Etiquetas HTML

```
<input type="submit" value="Enviar">
```

```
<input type="reset" value="Borrar">
```

```
<input type="button" value="Genérico">
```





JavaScript

$\approx \neq$



- Operadores
- Estructuras de control
  - Condiciones
  - Ciclos





JavaScript

Lenguaje script de la WEB,  
interpretado en tiempo de ejecución  
(en lugar de ser compilado).



Utilizado del lado del cliente



Se puede insertar en páginas HTML.

Compatible con múltiples navegadores



# Introduciendo JavaScript en documentos HTML

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <script>
      document.write("<p>This is my first line using JavaScript</p>");
    </script>
  </body>
</html>
```

Se pueden insertar scripts en el `<body>` o `<head>`



# Introduciendo JavaScript en documentos HTML

```
const all_parameters= window.location.search;
```

```
document.write("Estos son los parametros:"+ all_parameters)
```

```
const urlParams = new URLSearchParams(all_parameters);
```

```
const nombre = urlParams.get("nombre")
```

```
console.log(nombre);
```

```
document.write("Este el nombre:"+ nombre)
```

JAVASCRIPT



# Haciendo referencia a elementos de HTML

## DOM (Document Object Model)

- Estándar oficial de W3C para acceder a elementos HTML

```
document.write()
```

```
document.getElementById("id")
```

```
document.getElementById("id").innerHTML
```



# Variables



- Las **variables locales** se declaran con “**var**”

Para declarar una **variable global**, se omite “**var**”

```
const pi=3.14;  
var person="Juan Perez";
```

- Tipos dinámicos:** la misma variable puede almacenar diferentes tipos de datos

```
var x;           // Ahora x no está definida  
x = 5;          // Ahora x es un número  
x = "Juan";     // Ahora x es una String
```

- Arrays**

```
var cars=new Array();  
cars[0]="Saab";  
cars[1]="Volvo";  
cars[2]="BMW";
```

```
var cars=new Array("Saab","Volvo","BMW");
```

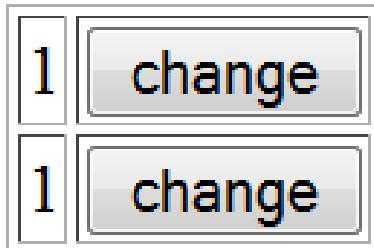
```
var cars=["Saab","Volvo","BMW"];
```

# Funciones



```
<script>
```

```
function changeText(elementId) {  
    var element=document.getElementById(elementId);  
    element.innerHTML="12345678910";  
}  
</script>
```



Normalmente las funciones se agregan en el head o al final del documento

```
function myFunction() {  
    var x=5;  
    return x;  
}
```

```
<input type="button" value="change" onclick="changeText('one');" />
```

# Reaccionando a eventos

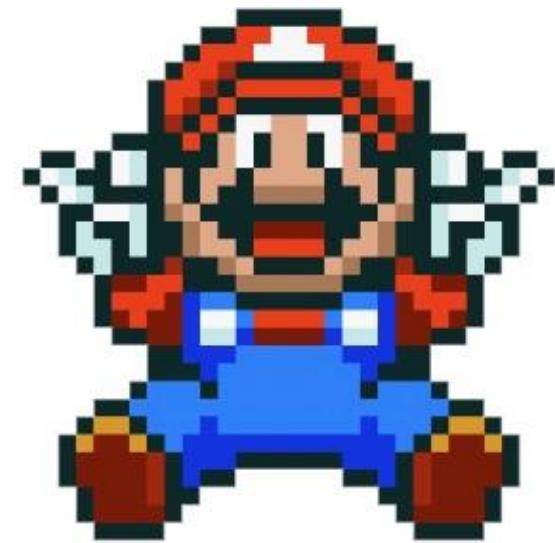
```
<script>
```

```
    function changelImage() {  
        element=document.getElementById('myimage')  
        if (element.src.match("down")) {  
            element.src="IMAGES/up.jpg";  
        } else {  
            element.src="IMAGES/down.jpg";  
        }  
    }  
</script>
```

Referencia a elemento  
Funciones en los eventos de los componentes

```

```



# JSON: Objetos en JavaScript



```
<p>
```

```
    Name: <span id="aName"> </span> <br/>
```

```
    Year: <span id="aYear"> </span> <br/>
```

```
</p>
```

```
<script>
```

```
var anObject= {  
    name:"Smith",  
    year:1984};
```

```
anObject.name;  
anObject["name"];
```

```
document.getElementById("aName").innerHTML=anObject.name;  
document.getElementById("aYear").innerHTML=anObject.year;
```

```
var person=new Object();  
person.firstname="Juan";  
person.age=50;  
document.write(person.firstname + " is " + person.age + " years old.");  
</script>
```

# JSON: Objetos en JavaScript

```
<script>
    var person=new Object();
    person.firstname="Juan";
    person.age=50;
    person.aMethod= function(x){
        person.age = person.age + x;
    }
    person.aMethod(3);
    document.write(person.firstname + " is " + person.age + " years old.");
</script>
```



# Excepciones en JavaScript

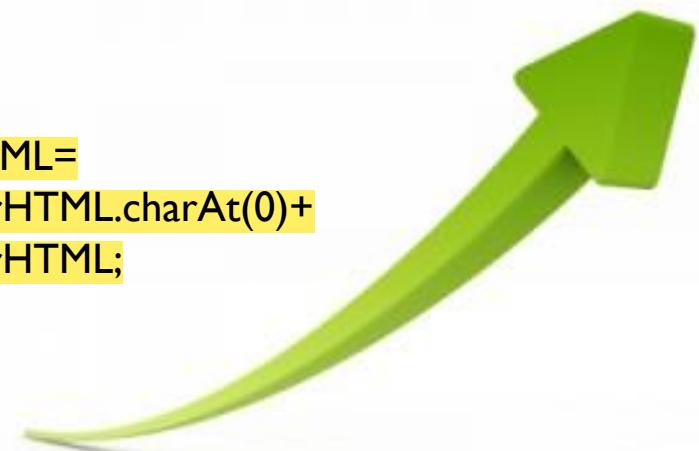
```
<script>
function myFunction() {
    var y=document.getElementById("mess");
    y.innerHTML="";
    try {
        var x=document.getElementById("demo").value;
        if(x=="")      throw "empty";
        if(isNaN(x))  throw "not a number";
        if(x>10)      throw "too high";
        if(x<5)       throw "too low";
    } catch(err) {
        y.innerHTML="Error: " + err + ".";
    }
}
</script>
```



```
<p>Please input a number between 5 and 10:</p>
<input id="demo" type="text">
<button type="button" onclick="myFunction()">Test Input</button>
<p id="mess"></p>
```

# Añadiendo eventos: Ejemplo I

```
<script>  
    function increaseText(id){  
        document.getElementById(id).innerHTML=  
            document.getElementById(id).innerHTML.charAt(0)+  
            document.getElementById(id).innerHTML;  
    }  
</script>
```

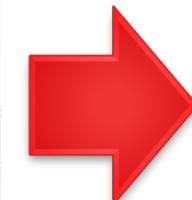


```
<h1 id ="anyId1" onclick="increaseText('anyId1')">Click on this text!</h1>  
<h1 id ="anyId2" onclick="increaseText('anyId2')">Click on this text!</h1>
```

# Añadiendo eventos: Ejemplo 2

```
<script>  
    function upperCase(id){  
        var text=document.getElementById(id);  
        text.value=text.value.toUpperCase();  
    }  
</script>
```

a	b	c
d	e	f
g	h	i
j	k	l
m	n	o
p	q	r
s	t	u
v	w	x
y	z	



A	B	C
D	E	F
G	H	I
J	K	L
M	N	O
P	Q	R
S	T	U
V	W	X
Y	Z	

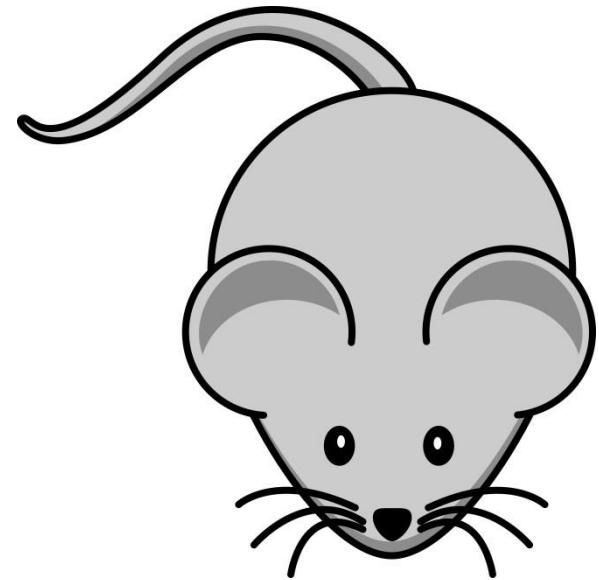
```
<input type="text" id="anyId3" onchange="upperCase('anyId3');">
```

# Añadiendo eventos: Ejemplo 3

```
<script>  
function eventMouseOver(obj) {  
    obj.innerHTML="Thank You";  
}  
</script>
```

```
function eventMouseOut(obj){  
    obj.innerHTML="Mouse Over Me";  
}  
</script>
```

```
<div onmouseover="eventMouseOver(this)"  
     onmouseout="eventMouseOut(this)">  
    Mouse Over Me  
</div>
```



# Validación de Formularios

```
<script>
function validateForm(){
    var x=document.forms["myForm"]["fname"].value;
    if (x==null || x=="") {
        alert("First name must be filled out");
        return false;
    }
    return true;
}
</script>
```

```
<form name="myForm"
      action="nextpage.html"
      onsubmit="return validateForm()"
      method="get">
```

First name: <input type="text" name="fname"> <br />

```
          <input type="submit" value="Submit">
</form>
```

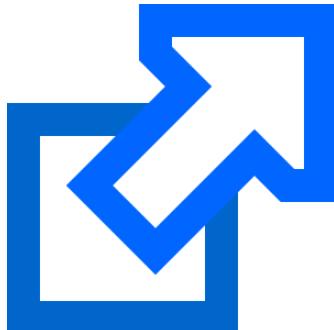


# Archivos externos de JavaScript

```
<html>
<body>

    <script src="myjavascriptfile.js"></script>
    <script src="myjavascriptfile.js" defer></script>
    <script src="myjavascriptfile.js" async></script>

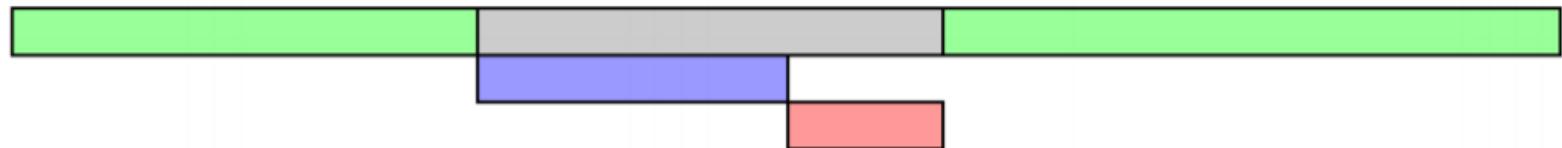
</body>
</html>
```



# Orden de Ejecución de Scripts

- Descarga y parseo de HTML
- Descarga y parseo de HTML detenidos
- Descarga del script
- Ejecución del script

`<script>`



`<script async>`



`<script defer>`



# Temporización



- **setTimeout(función, tiempo)**
  - Dispara la función una sola vez
  - clearTimeout() borra el evento
- **setInterval(función, tiempo)**
  - Dispara la función repetidamente
  - clearInterval() borra el evento

# Eventos de la “ventana”

- **onafterprint**
- **onbeforeprint**
- **onbeforeunload**
- **onerror**
- **onhaschange**
- **onload**
- **onmessage**
- **onoffline**
- **ononline**
- **onpagehide**
- **onpageshow**
- **onpopstate**
- **onredo**
- **onresize**
- **onstorage**
- **onundo**
- **onunload**



# Eventos de “Form”

- **onblur**
- **onchange**
- **oncontextmenu**
- **onfocus**
- **onformchange**
- **onforminput**
- **oninput**
- **oninvalid**
- **onreset**
- **onselect**
- **onsubmit**



# Eventos del teclado

- **onkeydown**
- **onkeypress**
- **onkeyup**



# Eventos del mouse



- onclick
- ondblclick
- ondrag
- ondragend
- ondragenter
- ondragleave
- ondragover
- ondragstart
- ondrop
- onmousedown
- onmousemove
- onmouseout
- onmouseover
- onmouseup
- onmousewheel
- onscroll

# Browser Object Model - BOM

- Objeto **windows**. Ejemplos:
  - window.innerHeight
  - window.moveTo()
  - window.resizeTo()
- Objeto **screen**. Ejemplos:
  - screen.pixelDepth
- Objeto **history**. Ejemplos:
  - history.back()
  - history.forward()



# Browser Object Model - BOM

- Objeto **navigator**. Ejemplos:
  - navigator.cookieEnabled
  - navigator.systemLanguage
- **Mensajes**. Ejemplos:
  - alert
  - prompt
  - confirm



# Práctica de laboratorio

Utiliza un archivo .js

## Sistema de Cotización de Seguros de Auto

Datos personales

Campo	Valor
Nombre:	Jose
Apellidos:	Octavio
Género:	<input checked="" type="radio"/> Masculino <input type="radio"/> Femenino
Edad:	55
Estado:	Aguascalientes <input type="button" value="▼"/>
Limpiar	Enviar

**Eventos:**  
onchange  
onkeypress  
onkeydown  
onkeyup

Gracias Sr. Jose Octavio por el interés en nuestros seguros

If () {  
}  
} else {  
}  
}

# Práctica de laboratorio: request-reply

Extrae parámetros con

```
const params= window.location.search;  
const urlParams = new URLSearchParams(params);  
const a_param = urlParams.get("a_param");
```

Guarda parámetros con  
<input type='hidden' ...>



Datos de Auto - Mozilla Firefox

Archivo Editar Ver Historial Marcadores Herramientas Ayuda

Datos de Auto

localhost:8084/Wek

## Sistema de Cotización de Seguros de Auto

Datos personales:

Estimado Sr. Octavio Gutierrez

Género: masculino

Edad: Desconocida

Estado: Jalisco

Datos del auto

Campo	Valor
Marca:	Chevrolet
Modelo:	2008
Placas:	122-HGZ
Limpiar	Enviar



# Práctica de laboratorio

**Num.ToString()  
parseInt(texto)  
parseFloat(texto)**

Cotización - Mozilla Firefox

Archivo Editar Ver Historial Marcadores Herramientas Ayuda

Cotización

localhost:8084/Wel juro c

## Cotización de seguro para Automóvil

Estimado Sr(a):Octavio Gutierrez  
En función a los datos proporcionados:  
Género: masculino  
Edad: Desconocida  
Estado: Jalisco  
Marca: Chevrolet  
Modelo: 2008  
Placas: 122-HGZ

La cotización de su seguro es:

**\$7,800.00 pesos**

Cotización - Mozilla Firefox

Archivo Editar Ver Historial Marcadores Herramientas Ayuda

Cotización

localhost:8084/Wel juro c

## Cotización de seguro para Automóvil

Estimado Sr(a):Octavio Gutierrez  
En función a los datos proporcionados:  
Género: masculino  
Edad: Desconocida  
Estado: Jalisco  
Marca: Chevrolet  
Modelo: 1999  
Placas: 122-HGZ

La cotización de su seguro es:

**\$4,200.00 pesos**

# django



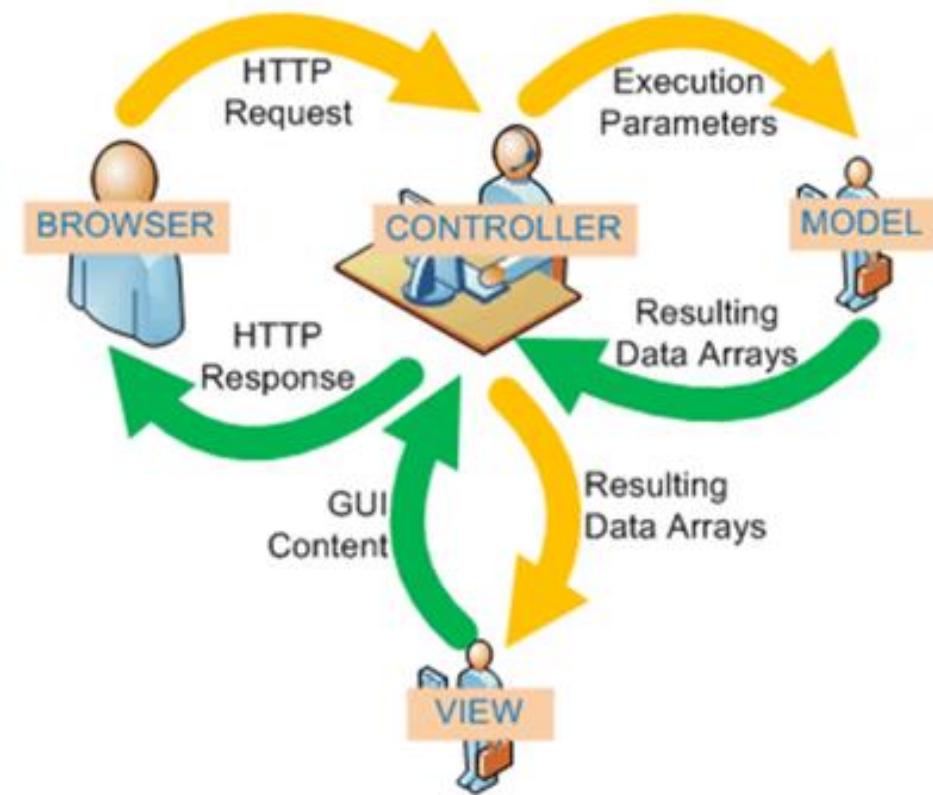
# Django

- Web framework escrito en Python
- Ventajas
  - Derivadas de Python
  - Aplicaciones plug & play
  - Mapeo Objeto Relacional: Objetos / Base de Datos
  - Lenguaje para plantillas
  - Interfaz de administrador (configurable)
  - Diseño de URLs “elegante”
  - MVC

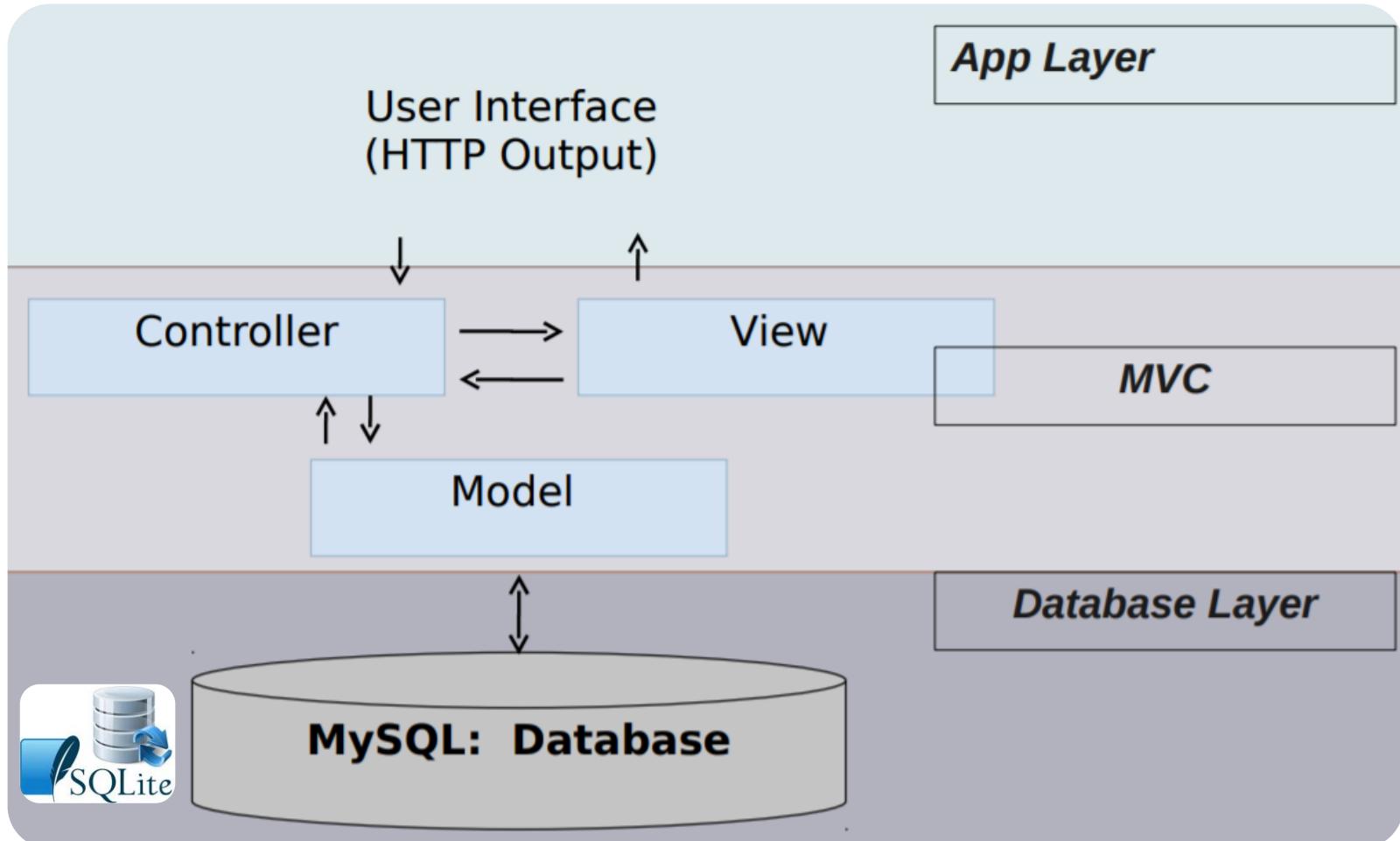


# Modelo Vista Controlador

- Datos
- Interfaz
- Lógica de control
  - Intermediario entre Datos e Interfaz



# django



# Sitio webs en Django

- Un sitio web en Django está conformado por una o múltiples apps:
  - Ingreso y registro de usuarios
  - Encuesta
  - Contacto
  - Carrito
  - Etc.



# Apps en Django

- Cada App tiene:
  - Modelo (datos)
  - Páginas web (vista)
- Cuando se crea una App, Django crea:
  - `models.py`
    - Almacena información de la App
  - `views.py`
    - Funciones de Python que crean lo que usuario ve.
  - `urls.py`

# Views

- Las funciones listadas en views.py tiene como entrada un objeto HttpRequest y como salida un objeto HttpResponse



# HttpRequest

- `HttpRequest.method`
- `HttpRequest.content_type`
- `HttpRequest.content_params`
- `HttpRequest.cookies`
- `HttpRequest.META`
- `HttpRequest.session`



# HttpResponse

- Es responsabilidad del programador crearla.
- `HttpResponse.content`
  - `HttpResponse.status_code`
  - `HttpResponse.set_cookie()`
- Subclase `JsonResponse`



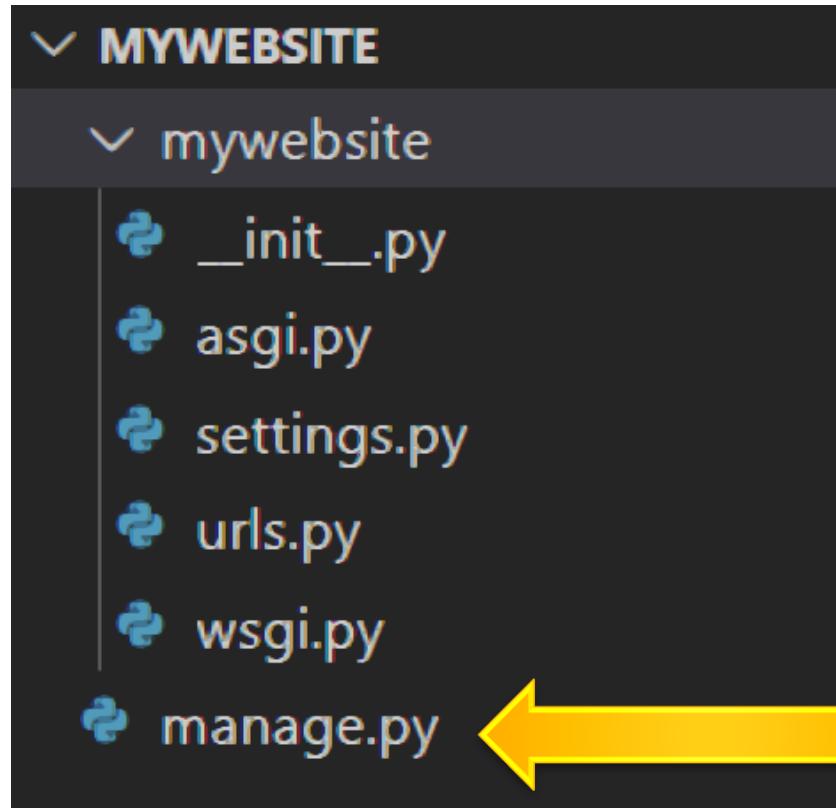
# Instalación de Django

- `python -m pip install Django`
- Anaconda package manager
- Verificar la instalación y versión
  - `python -m django --version`



# Creación de un proyecto

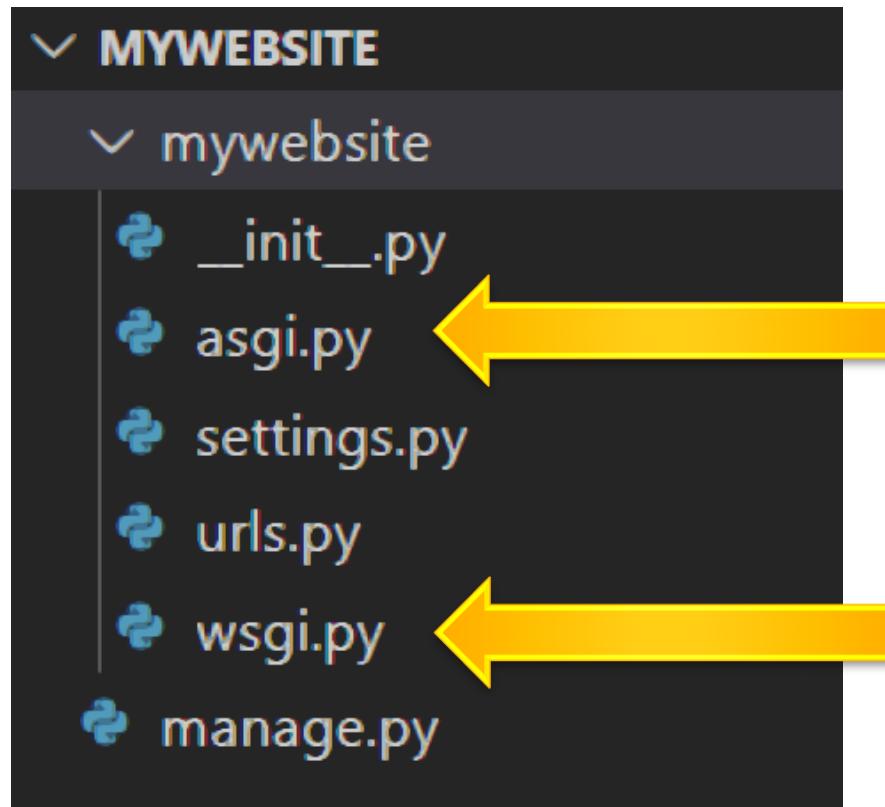
- `django-admin startproject mywebsite`



Permite  
interactuar con  
el proyecto  
Django

# Creación de un proyecto

- `django-admin startproject mywebsite`



Asynchronous  
Server Gateway  
Interface

Web Server  
Gateway  
Interface

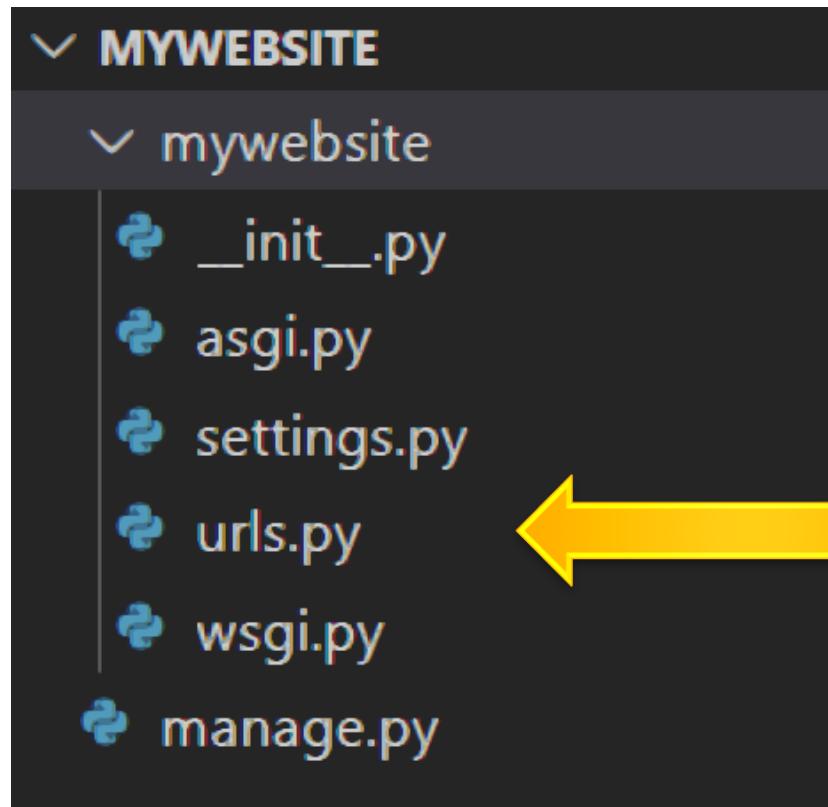
# Creación de un proyecto

- django-admin startproject mywebsite



# Creación de un proyecto

- django-admin startproject mywebsite



Índice de  
servicios/páginas  
ofrecidas

# Arranca el Servidor Web

...\\directory\\mywebsite

`python manage.py runserver`

`http://127.0.0.1:8000/`



The install worked successfully! Congratulations!

# Crea una App

- `python manage.py startapp myfirstapp`

```
MYWEBSITE
└── myfirstapp
    ├── migrations
    ├── __init__.py
    ├── admin.py
    ├── apps.py
    ├── models.py
    ├── tests.py
    └── views.py
    ├── mywebsite
    └── db.sqlite3
    └── manage.py
```

# Escribir una Vista

myfirstapp/views.py

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.

def index(request):
    return HttpResponse("Hola a todos desde myfirstapp")
```

# Vincula la Vista a una URL

Crea myfirstapp/urls.py

```
from django.urls import path  
  
from . import views  
  
urlpatterns = [  
    path('', views.index, name='index'),  
]
```

# Vincular URLs de la App con las de Sitio Web

En `mywebsite/urls.py`

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
    path('myfirstapp/', include('myfirstapp.urls')),
    path('admin/', admin.site.urls),
]
```

# Aplicaciones instaladas por defecto

...\\directory\\mywebsite\\settings.py

Crea las tablas de la BD de las aplicaciones

python manage.py migrate

Vuelve a ejecutar el servidor web

python manage.py runserver

# Crea Modelos: Dominio

Estudiante



Carrera



# Crea Modelos

.../myfirstapp/**models.py**

```
from django.db import models

class Estudiante(models.Model):
    nombre = models.CharField(max_length=200)
    apellidos = models.CharField(max_length=200)
    edad = models.IntegerField(default=0)
    promedio = models.FloatField(default=9.99)
    foraneo = models.BooleanField(default=False)

class Carrera(models.Model):
    LICENCIATURA = 1
    INGENIERIA = 2
    OPCIONES_TIPO = (
        (LICENCIATURA, 'Licenciatura'),
        (INGENIERIA, 'Ingenieria'),
    )
    estudiante = models.ForeignKey(Estudiante, on_delete=models.CASCADE)
    tipo = models.IntegerField(choices=OPCIONES_TIPO, null=True, blank=True)
    nombre = models.CharField(max_length=200)
```

# “Instala” tu aplicación

...\\directory\\mywebsite\\settings.py

```
INSTALLED_APPS = [  
    'myfirstapp.apps.MyfirstappConfig',  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
]
```

# Crea las tablas de tu aplicación recientemente “instalada”

- Para la simulación (preparar los archivos)

```
python manage.py makemigrations myfirstapp
```

- Échale un ojo a:

myfirstapp/migrations/0001\_initial.py

- Opcionalmente, échale el otro ojo a:

```
python manage.py sqlmigrate myfirstapp 0001
```

- Finalmente, crea las tablas.

```
python manage.py migrate
```

# Interactúa con el Shell de Django I

- `python manage.py shell`
- Importa los modelos

```
>> from myfirstapp.models import Estudiante, Carrera
```

```
>> Estudiante.objects.all()
```

- Error común al crear Objetos

```
>> estudiante = Estudiante ("José", "Gutiérrez", 49, 8.9, True)
```

```
>> estudiante.id
```

# Interactúa con el Shell de Django 2

- Forma correcta

```
>> estudiante = Estudiante (nombre="José", apellidos="Gutiérrez",  
edad=49, promedio=8.9, foraneo=True)
```

```
>> estudiante.id
```

```
>> estudiante.save()
```

```
>> estudiante.id
```

Añade otro estudiante

```
>>...
```

```
>> Estudiante.objects.all()
```

# Mostrando los Objetos “Correctamente”

.../myfirstapp/models.py

```
from django.db import models

class Estudiante(models.Model):
    ...
    def __str__(self):
        return self.nombre +" "+self.apellidos

    def aspira_a_beca(self):
        return self.promedio >=9

class Carrera(models.Model):
    ...
    def __str__(self):
        return str(self.tipo) +" "+self.nombre
```

# Interactúa con el Shell de Django 3

```
>> from myfirstapp.models import Estudiante, Carrera  
  
>> Estudiante.objects.all()  
  
>> estudiante = Estudiante.objects.get(id=1)  
>> estudiante = Estudiante.objects.get(pk=1)  
  
>> estudiante.carrera_set.all()  
  
>> estudiante.carrera_set.create(tipo=1, nombre="Matematicas")  
>> estudiante.carrera_set.create(tipo=2, nombre="Computacion")  
  
>> estudiante.carrera_set.all()
```

# Interactúa con el Shell de Django 4

```
>> carrera = estudiante.carrera_set.create(tipo=1,  
nombre="Actuaria")
```

```
>> carrera.estudiante
```

```
>> estudiante.carrera_set.count()
```

```
>> c = estudiante.carrera_set.get(id=1)
```

```
>> c.delete()
```

Agrega dos Carreras al estudiante 2

# Crea un Super usuario

- `python manage.py createsuperuser`

admin

admin@admin.com

123456

<http://127.0.0.1:8000/admin>

# Añade tu App a la Interfaz Admin

- .../myfirstapp/admin.py

```
from django.contrib import admin  
from .models import Estudiante, Carrera  
  
admin.site.register(Estudiante)  
admin.site.register(Carrera)
```

# Entra al sitio de Administración

- <http://127.0.0.1:8000/admin>
- Agrega un nuevo Estudiante

The screenshot shows the Django Admin interface with a dark theme. At the top, it says "Site administration". Below that is a blue header bar labeled "AUTHENTICATION AND AUTHORIZATION". Under this, there are two items: "Groups" with "Add" and "Change" buttons, and "Users" with "Add" and "Change" buttons. A horizontal line separates this from the next section. The next section has a blue header bar labeled "MYFIRSTAPP". It contains three items: "Carreras" with "Add" and "Change" buttons, and "Estudiantes" with "Add" and "Change" buttons. Each item has a small green plus icon next to the "Add" button and a yellow pencil icon next to the "Change" button.

AUTHENTICATION AND AUTHORIZATION		
Groups	<a href="#">+ Add</a>	<a href="#">Change</a>
Users	<a href="#">+ Add</a>	<a href="#">Change</a>

MYFIRSTAPP		
Carreras	<a href="#">+ Add</a>	<a href="#">Change</a>
Estudiantes	<a href="#">+ Add</a>	<a href="#">Change</a>

# Creando más vistas y paso de parámetros

- myfirstapp/views.py

```
...  
  
def detalles(request, estudiante_id):  
    return HttpResponse("Detalles del estudiante %s." % estudiante_id)  
  
def carreras(request, estudiante_id):  
    return HttpResponse("Carreras del estudiante %s." % estudiante_id)  
  
def agrega_carrera(request, estudiante_id):  
    return HttpResponse("Agregando una carrera a %s." % estudiante_id)
```

# Vinculando las vistas a URLs

- myfirstapp/urls.py

...

```
urlpatterns = [  
    ...  
  
    # ejemplo: /myfirstapp/5/  
    path('<int:estudiante_id>', views.detalles, name='detalles'),  
  
    # ejemplo: /myfirstapp/5/carreras/  
    path('<int:estudiante_id>/carreras/', views.carreras, name='carreras'),  
  
    # ejemplo: /myfirstapp/5/agrega_carrera/  
    path('<int:estudiante_id>/agrega_carrera/', views.agrega_carrera, name='agrega_carrera'),  
]  
]
```

## Convertidores:

- int
- str
- slug: ascii con \_ -

# Aumentando funcionalidad de las Views

- myfirstapp/views.py

```
...
from .models import Estudiante
...
def index(request):
    estudiantes = Estudiante.objects.order_by('nombre')
    lista = ""
    for e in estudiantes:
        lista += e.__str__()+" "
    return HttpResponse(lista)
...
```

# El lenguaje de template de Django

```
{% if athlete_list %}  
    Number of athletes: {{ athlete_list|length }}  
{% elif athlete_in_locker_room_list %}  
    Athletes should be out of the locker room soon!  
{% else %}  
    No athletes.  
{% endif %}
```

- <https://docs.djangoproject.com/en/3.2/ref/templates/language/>

# Creando Templates

- myfirstapp/templates/myfirstapp/index.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Index page</title>
  </head>
  <body>
    <p>Listado de estudiantes</p>
    {% if estudiantes %}
      <ul>
        {% for estudiante in estudiantes %}
          <li><a href="/myfirstapp/{{ estudiante.id }}/detalles">
            {{estudiante.nombre}}
            {{estudiante.apellidos}}
          </a></li>
        {% endfor %}
      </ul>
      {% else %}
        <p>No hay estudiantes registrados.</p>
      {% endif %}
    </body>
  </html>
```

# Revisitando la vista index I

- myfirstapp/views.py

```
...  
  
from django.template import loader  
  
...  
  
def index(request):  
    estudiantes = Estudiante.objects.order_by('nombre')  
    template = loader.get_template('myfirstapp/index.html')  
    context = {  
        'estudiantes': estudiantes,  
    }  
    return HttpResponseRedirect(template.render(context, request))  
  
...
```

# Revisando la vista index 2

- myfirstapp/views.py

```
...  
from django.shortcuts import render  
  
...  
  
def index(request):  
    estudiantes = Estudiante.objects.order_by('nombre')  
    context = {  
        'estudiantes': estudiantes,  
    }  
    return render(request, 'myfirstapp/index.html', context)  
  
...
```

# Creando template para detalles

- templates/myfirstapp/detalles.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Detalles</title>
  </head>
  <body>
    <p> Información del Estudiante </p>
    {% if estudiante %}
      <p>{{ estudiante.id }} </p>
      <p>{{ estudiante.nombre }}</p>
      <p>{{ estudiante.apellidos }} </p>
      <p>{{ estudiante.promedio }} </p>
      {% if estudiante.foraneo %}
        <p>Foráneo</p>
      {% else %}
        <p>Local </p>
      {% endif %}
    {% endif %}
  </body>
</html>
```

# Actualizando la vista Detalles

- myfirstapp/views.py

```
...  
  
def detalles(request, estudiante_id):  
    estudiante = Estudiante.objects.get(pk=estudiante_id)  
    return render(request, 'myfirstapp/detalles.html', {'estudiante': est  
udiante})  
  
...
```

# Excepciones HTTP

- myfirstapp/views.py

```
...  
  
from django.http import Http404  
  
...  
  
def detalles(request, estudiante_id):  
    try:  
        estudiante = Estudiante.objects.get(pk=estudiante_id)  
    except Estudiante.DoesNotExist:  
        raise Http404()  
    return render(request, 'myfirstapp/detalles.html',  
                 {'estudiante': estudiante})  
  
...
```

# Simplificando la excepción HTTP

- myfirstapp/views.py

```
...  
from django.shortcuts import get_object_or_404  
  
...  
  
def detalles(request, estudiante_id):  
    estudiante = get_object_or_404(Estudiante, pk=estudiante_id)  
    return render(request, 'myfirstapp/detalles.html',  
                  {'estudiante': estudiante})  
  
...
```

# Mostrando las Carreras en el Template detalles.html [1/3]

- template/detalles.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Página de detalles</title>
    </head>
    <body>
        <p> Información del Estudiante </p>
        ...
        <ul>
            {% for carrera in estudiante.carrera_set.all %}
                <li> {{ carrera.tipo }} {{ carrera.nombre }} </li>
            {% endfor %}
        </ul>
        {% endif %}
    </body>
</html>
```

# Mostrando las Carreras en el Template detalles.html [2/3]

- myfirstapp/models.py

```
class Carrera (models.Model):  
    LICENCIATURA = 1  
    INGENIERIA = 2  
    OPCIONES_TIPO = ( (LICENCIATURA, "Licenciatura"),  
                      (INGENIERIA, "Ingenieria") )  
  
    estudiante = models.ForeignKey(Estudiante, on_delete=models.CASCADE)  
    tipo = models.IntegerField(choices=OPCIONES_TIPO, null=True, blank=True)  
    nombre = models.CharField(max_length=100)  
  
    ...  
  
    def get_tipo(self):  
        return Carrera.OPCIONES_TIPO[self.tipo-1][1]
```

# Mostrando las Carreras en el Template detalles.html [3/3]

- template/detalles.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Página de detalles</title>
    </head>
    <body>
        <p> Información del Estudiante </p>
        ...
        <ul>
            {% for carrera in estudiante.carrera_set.all %}
                <li> {{ carrera.get_tipo }} {{ carrera.nombre }} </li>
            {% endfor %}
        </ul>
        {% endif %}
    </body>
</html>
```

# Agregando un Namespace a la App

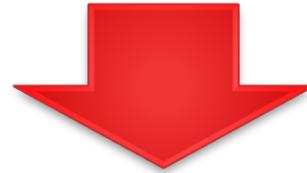
- myfirstapp/urls.py

```
from django.urls import path  
  
from . import views  
  
app_name = 'myfirstapp'  
  
urlpatterns = [  
    path('', views.index, name='index'),  
  
    ...  
]
```

# Quitando URL cableadas en Templates

- myfirstapp/templates/myfirstapp/index.html

```
<li><a href="/myfirstapp/{{estudiante.id}}/detalles">
    {{estudiante.nombre}}
    {{estudiante.apellidos}}
</a></li>
```



```
<li><a href="{% url 'myfirstapp:detalles' estudiante.id %}">
    {{estudiante.nombre}}
    {{estudiante.apellidos}}
</a></li>
```

# Agrega una carrera

```
def agrega_carrera(request, estudiante_id, tipo, nombre):  
    estudiante = Estudiante.objects.get(pk=estudiante_id)  
    estudiante.carrera_set.create(tipo = int(tipo), nombre=nombre)  
    return HttpResponseRedirect("Carrera agregada al estudiante %s" % estudiante_id)
```

# Agrega un Estudiante

```
def agrega_estudiante(request, nombre, apellidos, edad, foraneo, promedio):
    foraneo = "True".lower() == "true"
    estudiante = Estudiante(
        nombre=nombre,
        apellidos=apellidos,
        edad=int(edad),
        foraneo=foraneo,
        promedio=float(promedio))

    estudiante.save()
    return HttpResponse("Estudiante %s agregado exitósamente" % estudiante.id
)
```

# Borra un Estudiante

```
def borra_estudiante(request, estudiante_id):
    estudiante = Estudiante.objects.get(pk=estudiante_id)
    estudiante.delete()
    return HttpResponse("Estudiante %s borrado exitósamente" % estudiante_id)
```

# Edita un Estudiante

```
def edita_estudiante(request, estudiante_id, promedio):
    estudiante = Estudiante.objects.get(pk=estudiante_id)
    estudiante.promedio = float(promedio)
    estudiante.save()
    return HttpResponse("El promedio del estudiante %s se ha actualizado exitosamente" % estudiante.id)
```

# Agrega un Estudiante – POST [I/2]

```
def agrega_estudiante_forma(request):
    nombre = request.POST.get("nombre")
    apellidos = request.POST.get("apellidos")
    edad = int(request.POST.get("edad"))
    if request.POST.get("foraneo") == None:
        foraneo = False
    else:
        foraneo = request.POST.get("foraneo").lower() == "true"
    promedio = float(request.POST.get("promedio"))
    estudiante = Estudiante(      nombre=nombre,
                                  apellidos=apellidos,
                                  edad=int(edad),
                                  foraneo=foraneo,
                                  promedio=float(promedio))
    estudiante.save()
    return HttpResponse("Estudiante %s agregado exitósamente desde la forma"
% estudiante.id)
```

# Agrega un Estudiante – POST [I/2]

```
<form action="agrega_estudiante_forma" method="POST">
    {%csrf_token%}
    <label> Nombre: <input type="text" name = "nombre"> </label> <br>
    <label> Apellidos: <input type="text" name = "apellidos"> </label><br>
    <label> Edad: <input type="text" name = "edad"> </label><br>
    <label> Promedio: <input type="text" name = "promedio"> </label><br>
    <label> <input type="checkbox" value="True" name="foraneo"> Foráneo </label><br>
    <input type="submit" value="Aregar">
</form>
```

# URLs de Alta, Bajas y Ediciones

```
urlpatterns = [  
  
    path(' ', views.index, name="index"),  
    path('<int:estudiante_id>/detalles/', views.detalles, name="detalles"),  
    path('<int:estudiante_id>/<int:tipo>/<str:nombre>/agrega_carrera/', views.agrega_carrera, name="agrega_carrera"),  
    path('<str:nombre>/<str:apellidos>/<int:edad>/<str:foraneo>/<str:promedio>/agrega_estudiante/', views.agrega_estu  
diante, name="agrega_estudiante"),  
    path('<int:estudiante_id>/<str:promedio>/edita_estudiante/', views.edita_estudiante, name="edita_estudiante"),  
    path('<int:estudiante_id>/borra_estudiante/', views.borra_estudiante, name="borra_estudiante"),  
    path('agrega_estudiante_forma', views.agrega_estudiante_forma, name="agrega_estudiante_forma"),  
]
```

# Incluyendo CSS

- static/myfirstapp/style.css

```
p {  
    color: red;  
    text-align: center;  
}
```

```
<head>  
    <meta charset="utf-8">  
    <title>Este es nuestro index del template</title>  
    {% load static %}  
    <link rel="stylesheet"  
          type="text/css"  
          href="{% static 'myfirstapp/style.css' %}">  
  
</head>
```

# CSS - Cascading Style Sheets

**#id**

**.class**

**element.class**

**\***

**element**

**element, element, ..**