UNIVERSIDAD NACIONAL DE TRUJILLO

INGENIERIA DE SOFTWARE I

INTEGRANTES:

A Azañedo Gamez, Wilson David.

B Hernández Barba, Marcos Ivan.

C Muñoz Vargas, José Antonio.

D Villanueva Briceño, Arturo Eduardo.

ÍNDICE

Manual de CRUD de Categorías

1. Listado con Paginación

- En view.py colocamos:

```
def listarcategoriaMia(request):
    queryset=request.GET.get("buscar")
    categorias=Categoria.objects.raw('SELECT * FROM "ventasApp_categoria"
 WHERE estado=True')
    if queryset:
        ToF=True
        filtro='%{}%'.format(queryset)
        categorias=Categoria.objects.raw('SELECT * FROM "ventasApp_catego
ria" WHERE estado=%s and descripcion LIKE %s',[ToF,filtro])

    paginator = Paginator(categorias, 3) #Pagina el listado de categorias
 de 3 en 3
    numero_pagina = request.GET.get('page') or 1 #Obtiene la página actua
l en la que se está, pero si aún no se pulsa una página en un comienzo en
tonces por defecto es primera página
    categorias = paginator.get_page(numero_pagina) #Obtiene solo las cate
gorías que pertenecen a tal página
    pagina_actual= int(numero_pagina) #Se castea la página actual, se
convierte a entero
    paginas=range(1, categorias.paginator.num_pages + 1) #el range siempr
e rescata desde el inicial hasta el (último-1) --> range(1,9) --
> rescata desde 1 hasta 8

    context={
        'categorias':categorias,
        'paginas': paginas,
        'pagina_actual':pagina_actual
    }
    return render(request,"categoriaMia/listarMia.html",context)
```

- **Modelo.objects.raw():** nos va a permitir SOLO realizar consultas directamente utilizando código SQL, en este caso con sintaxis de PostgreSQL. No permite utilizar el UPDATE, DELETE.
  Según documentación DJANGO, se utiliza de la siguiente manera:

```
>>> lname = 'Doe'

>>> Person.objects.raw('SELECT * FROM myapp_person WHERE last_name =
                       %s', [lname])
```

- **Paginación:** el detalle de la paginación se encuentra comentado en el código. Además, se tiene que importar:

```
from django.core.paginator import Paginator #Paginador que nos ofrece Dja
ngo
```

- En paginación.html colocamos:

```
<div class="row mt-3 justify-content-center">
    <nav>
        <ul class="pagination">
            {% if categorias.has_previous %} <!-
- si es que tenemos categorias en las páginas anteriores a la página en l
as que estamos -->
                <li class="page-item">
                    <a class="page-link" href="?page=1">Primera</a>
                </li>
                <li class="page-item">
                    <a class="page-
link" href="?page={{ categorias.previous_page_number }}">&laquo;</a> <!-
- categorias.previous_page_number hace que se retorne a la página anterio
r -->
                </li>
            {% endif %}

           <!--  <span class="current">
                Página {{ categorias.number }} de {{ categorias.paginator
.num_pages }}.
            </span> -->

            {% for pagina in paginas %}
                <li class="page-
item {% if pagina_actual == pagina %} active {% endif %}"> <!-
- si la página actual en la que estamos es igual a la página de la iterac
ion entonces permanece activo -->
                    <a class="page-
link" href="?page={{pagina}}">{{pagina}}</a>
                </li>
            {% endfor %}

            {% if categorias.has_next %} <!-
- si es que tenemos categorias en las páginas siguientes a la página en l
as que estamos -->
                <li class="page-item">
                    <a class="page-
link" href="?page={{ categorias.next_page_number }}">&raquo;</a>
```

```
                </li>
                <li class="page-item">
                    <a class="page-
link" href="?page={{ categorias.paginator.num_pages }}">Última</a> <!-
- categorias.paginator.num_pages hace ir a la última página -->
                </li>
            {% endif %}
        </ul>
    </nav>
</div>
```

- ** :** permite que se recupere la página actual por método GET en la vista.

- En listarMia.html colocamos:

```
{% extends "plantilla.html" %}
{% block content %}
<div class="container">
    <div class="card card-outline card-info mt-3">
        <div class="card-header">
        <h1 class="card-title">LISTADO DE CATEGORIAS</h1>
        <a href="{% url 'agregarcategoriaMia' %}" class='btn btn-sm btn-
primary btnadd'><i class='fas fa-plus'></i> NUEVO</a>
        </div>
    <div class="card-body">

        <div class="row">
            <div class="col-md-6 pull-right">
                <form action="" method="GET">
                <div class="input-group">
                    <input type="text" name="buscar" id="buscar" class="f
orm-control" autofocus placeholder="Buscar" value="">
                    <span class="input-group-btn">
                        <button class="btn btn-primary">
                        <i class="fa fa-search"></i>
                        </button>
                    </span>
                </div>
                </form>
            </div>
        </div>

        <table class="table mt-2">
            <table id="tablasubfamilias" class="table table-
striped mt4" style="width:100%">
                <thead class="bg-info text-white">
                <tr>
```

```
                <th>CODIGO</th>
                <th>DESCRIPCION</th>

                <th><center>OPCIONES</center></th>
                </tr>
                </thead>

            {% if categorias %}
                {% for itemcategoria in categorias %}
                    <tr>
                    <td>{{ itemcategoria.id }}</td>
                    <td>{{ itemcategoria.descripcion }}</td>

                    <td style="text-align: center;">
                    <a href="{% url 'editarcategoriaMia' itemcategoria.id
 %}" class="btn btn-info btnsm"><i class="fa fa-edit"></i> Editar</a>
                    <a href="#" onclick="eliminarCategoria('{{itemcategor
ia.id}}','{{itemcategoria.descripcion}}')" class="btn btn-danger btn-
sm"><i class="fa fatrash"></i> Eliminar</a>
                    </td>
                    </tr>
                {% endfor %}

                {% include "paginacion.html" %} <!-
- Incluye "paginacion.html" -->

            {% else %}
                <tr style="text-align: center;">
                <td colspan="4">
                <p>No hay Registros</p>
                </td>
                </tr>
            {% endif %}
            </table>


        </div>
        </div>
</div>

   {% endblock %}

   {% block js %}
        <script src="//cdn.jsdelivr.net/npm/sweetalert2@11"></script>
        {% if messages %}
        {% for m in messages %}
        <script>
        Swal.fire({
        "title": "Eliminado",
```

```
        "text":"{{m}}",
        "icon":"success"
        })
        </script>
        {% endfor %}
        {% endif %}
        <script>
        function eliminarCategoria(id,descripcion){
        Swal.fire({
        "title":"¿Realmente desea eliminar Registro?",
        "text":"Codigo :"+ id + " Descripción :" + descripcion,
        "icon":"question",
        "showCancelButton":true,
        "cancelButtonText":"No",
        "confirmButtonText":"Si",
        "reverseButton":true,
        "confirmButtonColor":"#dc3545"
        })
        .then(function(result){
        if(result.isConfirmed){
        window.location.href="/eliminarcategoriaMia/"+id+"/"
        }
        })
        }
        </script>
    {% endblock %}
```

- En urls.py:

```
 path('listarcategoriaMia/',listarcategoriaMia,name="listarcategoriaMia")
,
```

- Capturas del listado de categorías con paginación:

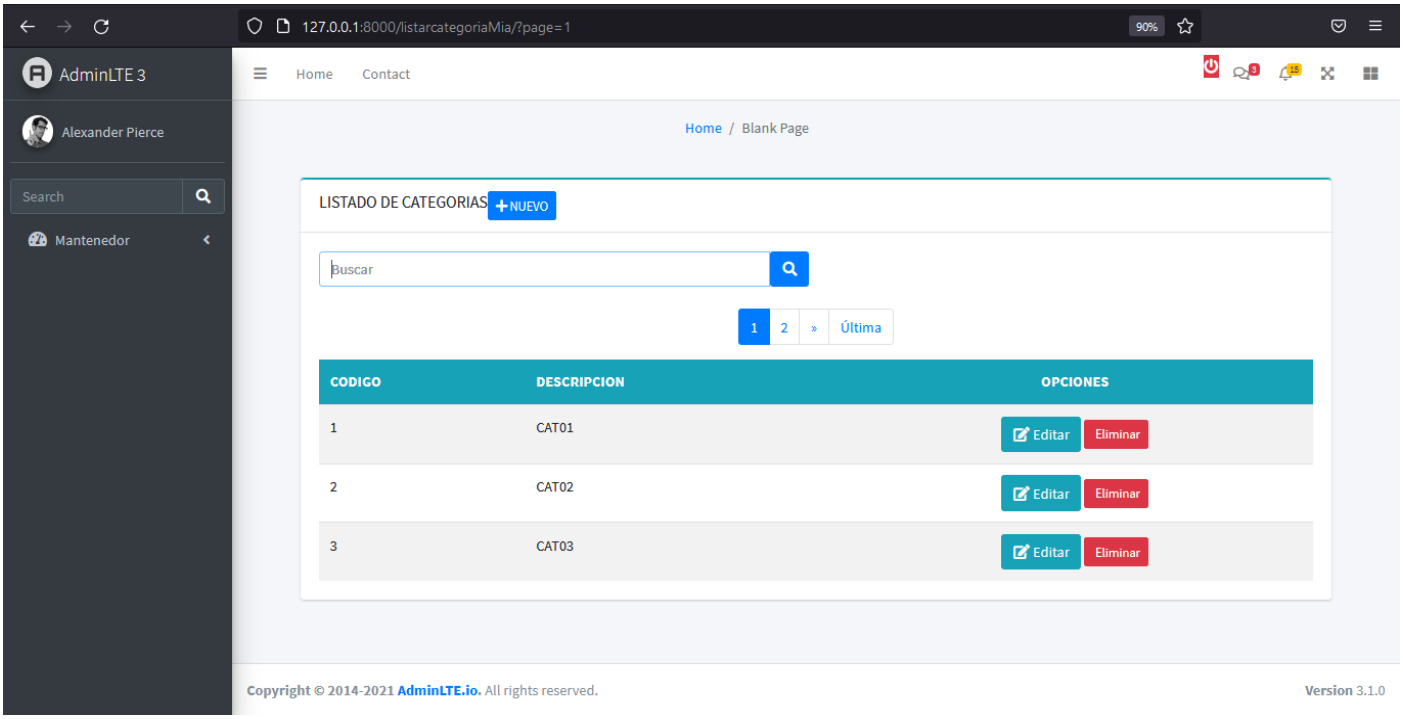

Figura 1: Listado de categorías con paginación – página 01

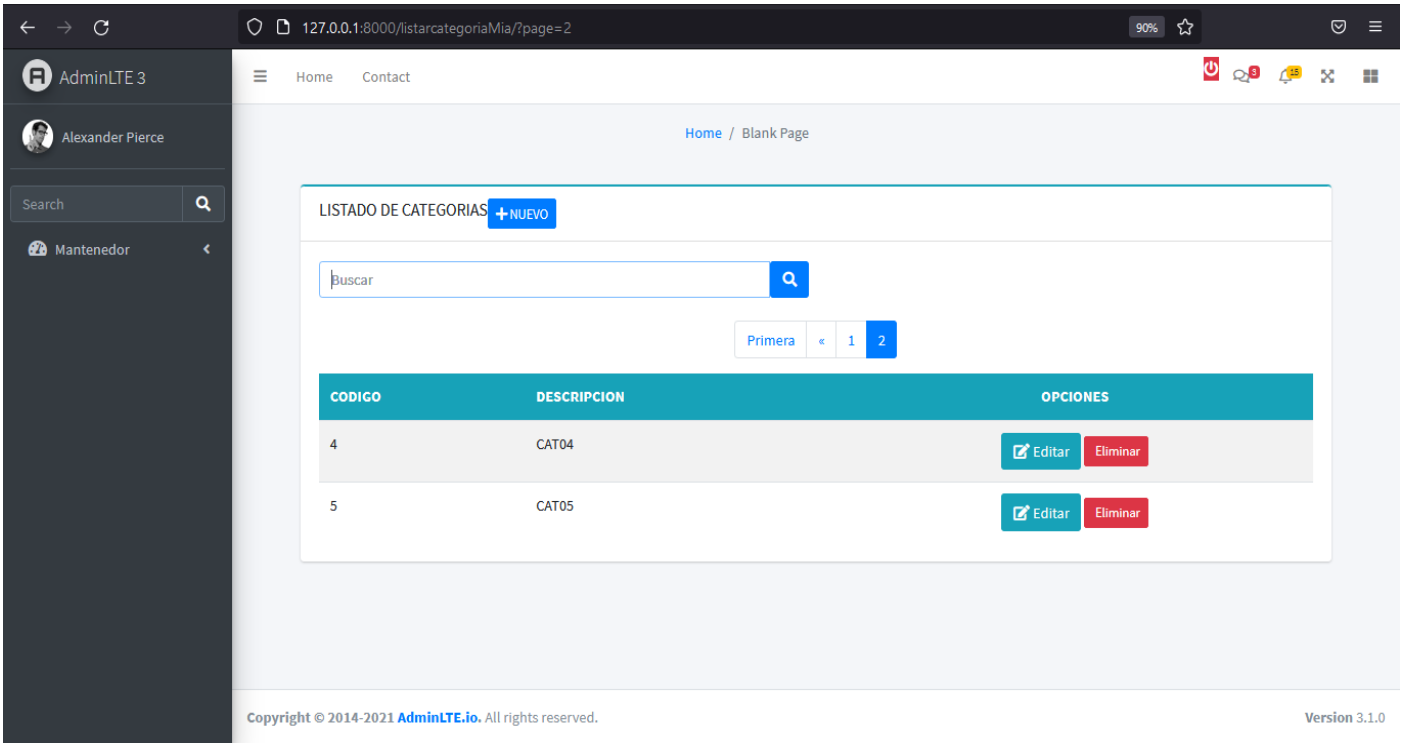

Figura 2: Listado de categorías con paginación – página 02

2. Agregar

- En view.py colocamos:

```
def agregarcategoriaMia(request):
    if request.method=="POST":
        crearCategoria= Categoria() #Instaciamos el modelo o clase Categoria()
        crearCategoria.descripcion=request.POST['descripcion']
        casillaEstado=request.POST.get('estado', 'off') #Si existe un request.POST['estado'] será igual a su valor que se le envió por el formulario, si no, será igual a 'off'.
        if casillaEstado=='on':
            crearCategoria.estado=True
        else:
            crearCategoria.estado=False
        crearCategoria.save() #Permite guardar el objeto en la base de datos
        return redirect("listarcategoriaMia")

    return render(request,"categoriaMia/agregarMia.html")
```

- Si se detecta un método "POST" entonces se procederá a guardar la información de la categoría instanciándola y recuperando la información por request.POST['nombre_objeto_formulario'], hay que tener cuidado con:

```
<input type="checkbox"  name="estado">
```

Porque cuando no se le selecciona no se envía ni el objeto ni la información del objeto, entonces para esto utilizamos el request.POST.get('nombre_objeto_formulario', 'default') → Si existe un request.POST['estado'] será igual a su valor que se le envió por el formulario, si no, será igual a 'off'. Además, tener en cuenta que si se selecciona el checkbox el valor será 'on', es por eso que después se trata con condicionales, para darle su valor de True o False según corresponda.

- En agregarMia.html colocamos:

```
{% extends "plantilla.html" %}

{% block content %}
    <div class="container">
        <div class="card card-info mx-4 my-4">
            <div class="card-header">
                <h1 class="card-title">AGREGAR CATEGORIA</h1>
            </div>
            <div class="card-body">
                <form method="post">
                    {% csrf_token %}
```

```
                    <div class="mb-3">
                        <label class="form-label">Descripción</label>
                        <input type="text" name="descripcion">
                    </div>
                    <div class="mb-3 form-check">
                        <input type="checkbox"  name="estado">
                        <label class="form-check-
label" for="exampleCheck1">Estado</label>
                    </div>
                    <div class="d-flex justify-content-end">
                    <a href="{% url 'listarcategoriaMia' %}" class="btn b
tn-secondary">Cancelar</a>
                     
                    <button type="submit" class="btn btnprimary">Guardar<
/button>
                    </div>
                </form>
            </div>
        </div>
    </div>
{% endblock%}
```

- En urls.py:

```
 path('agregarcategoriaMia/',agregarcategoriaMia ,name="agregarcategoriaM
ia"),
```

- Capturas sobre agregar categoría:

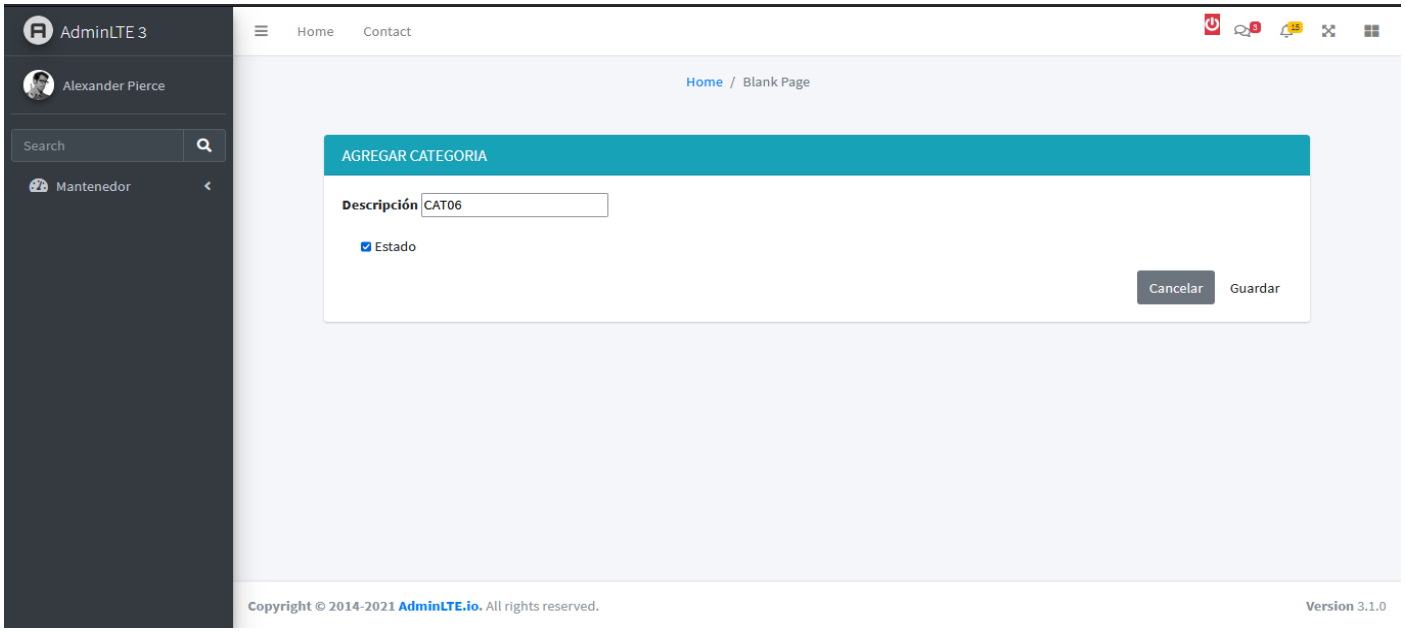

Figura 3: Agregando CAT06 con estado activado

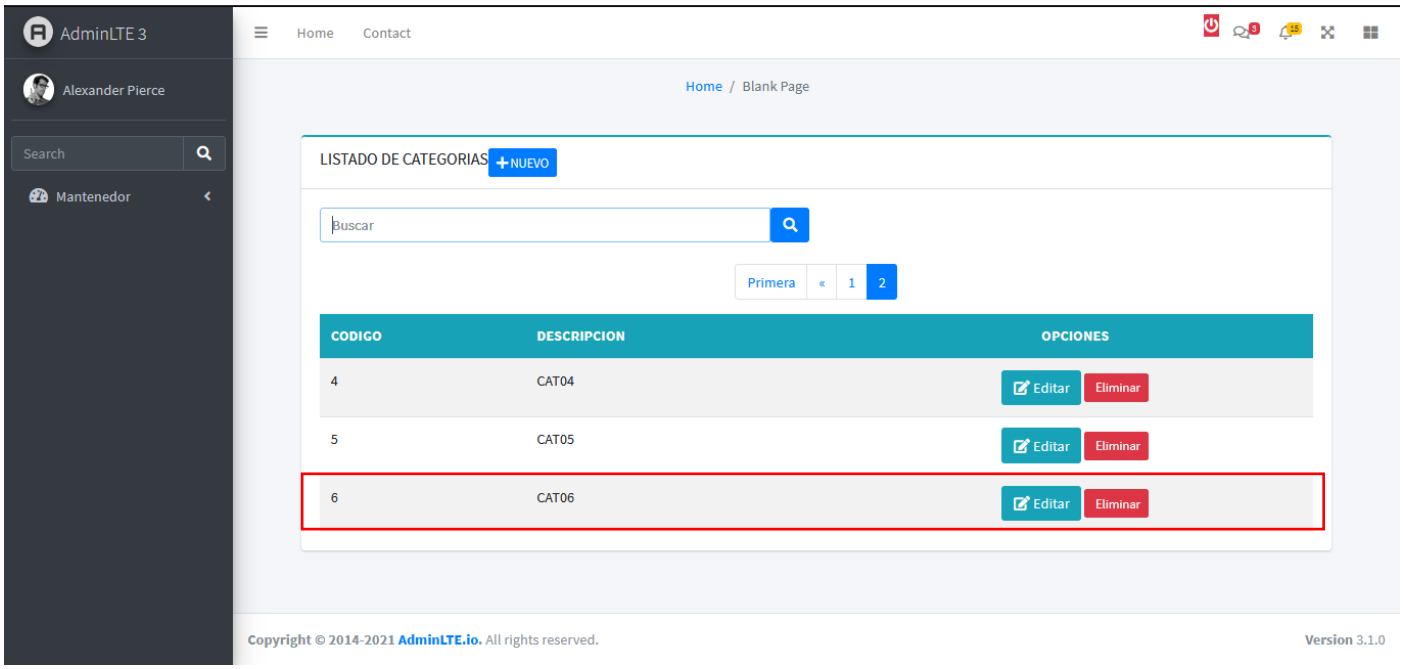

Figura 4: CAT06 con estado activado

3. Editar

- En view.py colocamos:

```
def editarcategoriaMia(request,id):
    if request.method=="POST":
        descripcionPost=request.POST['descripcion']
        casillaEstado=request.POST.get('estado', 'off') #Si existe un req
uest.POST['estado'] será igual a su valor que se le envió por el formular
io, si no, será igual a 'off'.
        if casillaEstado=='on':
            estadoPost=True
        else:
            estadoPost=False
        Categoria.objects.filter(id=id).update(descripcion=descripcionPos
t,estado=estadoPost)
        return redirect("listarcategoriaMia")
    else:
        categoria=Categoria.objects.raw('SELECT * FROM "ventasApp_categor
ia" WHERE id=%s LIMIT 1',[id])[0]
    context={"categoria":categoria}
    #print(categoria.estado)
    return render(request,"categoriaMia/editarMia.html",context)
```

- Si se detecta un método “POST”, es decir, se está intentando actualizar, utilizamos el ORM de Django:

```
Categoria.objects.filter(id=id).update(descripcion=descripcionPost,estado
=estadoPost)
```

Para obtener la Categoría con tal id, y después actualizar los campos descripción y estado de tal Categoría.

- Si se detecta un método "GET", es decir, se está intentando mostrar la información de la categoría que se seleccionó para editar, aquí nuevamente se utiliza la realización de consultas sin procesar:

```
 categoria=Categoria.objects.raw('SELECT * FROM "ventasApp_categoria" WHE
RE id=%s LIMIT 1',[id])[0]
```

Obtiene solo la categoría con tal id, limitándolo a uno en la búsqueda para obtener el primero que se obtenga.

- En editarMia.html colocamos:

```
{% extends "plantilla.html" %}
{% block content %}
    <div class="container">
        <div class="card card-info mx-4 my-4">
            <div class="card-header">
                <h1 class="card-title">EDITAR CATEGORIA</h1>
            </div>
            <div class="card-body">
                <form method="post">
                    {% csrf_token %}

                    <div class="mb-3">
                        <label class="form-label">Id: </label>
                        <input type="text" name="descripcion" value="{{ca
tegoria.id}}" disabled>
                    </div>
                    <div class="mb-3">
                        <label class="form-label">Descripción: </label>
                        <input type="text" name="descripcion" value="{{ca
tegoria.descripcion}}">
                    </div>
                    <div class="mb-3 form-check">

                            <input type="checkbox"  name="estado" {% if c
ategoria.estado == True %} checked {% endif %}>

                        <label class="form-check-
label" for="exampleCheck1">Estado</label>
                    </div>
                    <div class="d-flex justify-content-end">
                    <a href="{% url 'listarcategoriaMia' %}" class="btn b
tn-secondary">Cancelar</a>
```

```
                     
                    <button type="submit" class="btn btnprimary">Guardar<
/button>
                    </div>
                </form>
            </div>
        </div>
    </div>
{% endblock%}
```

- Simplemente en los value de los input se coloca el valor que se está enviando desde la vista, analicemos:

```
 <input type="checkbox"  name="estado" {% if categoria.estado == True %}
checked {% endif %}>
```

Va a permitir que se active el checkbox siempre y cuando tenga un estado True.

- En urls.py:

```
path('editarcategoriaMia/<int:id>/',editarcategoriaMia ,name="editarcateg
oriaMia"),
```

- Capturas sobre edición de categoría:

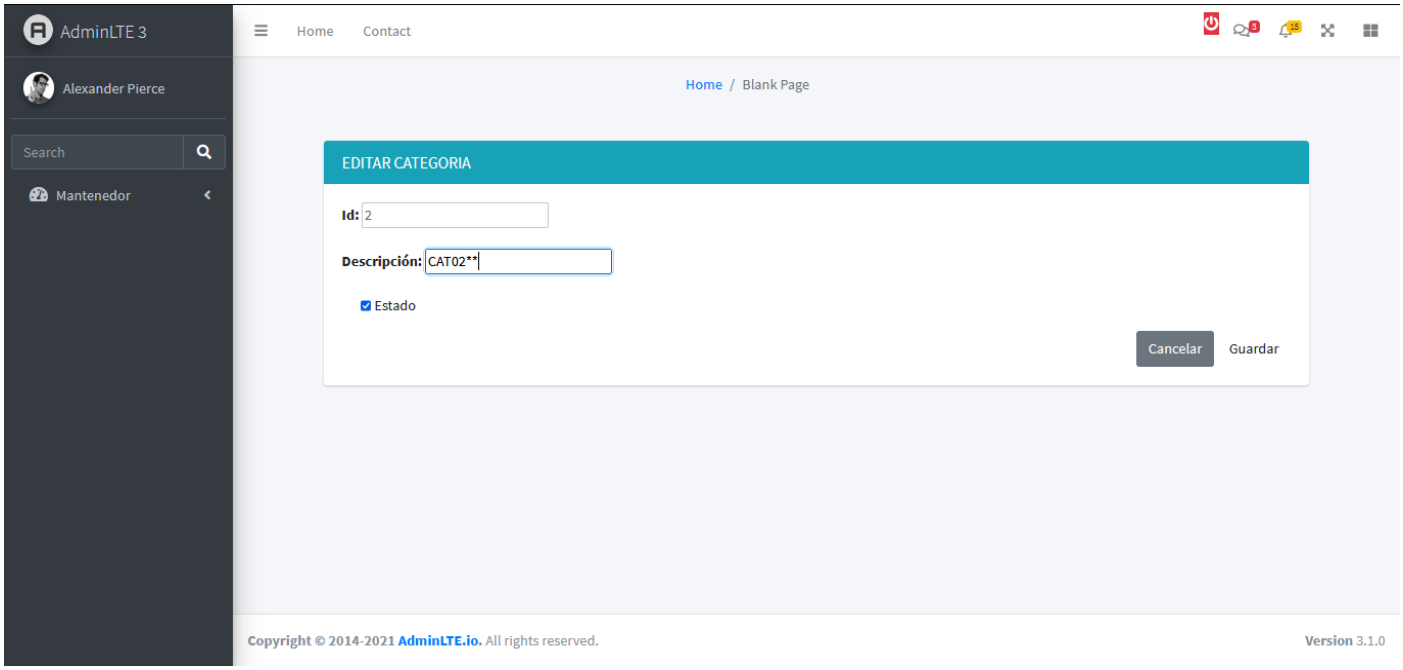

*Figura 5: Editar CAT02 a CAT02***

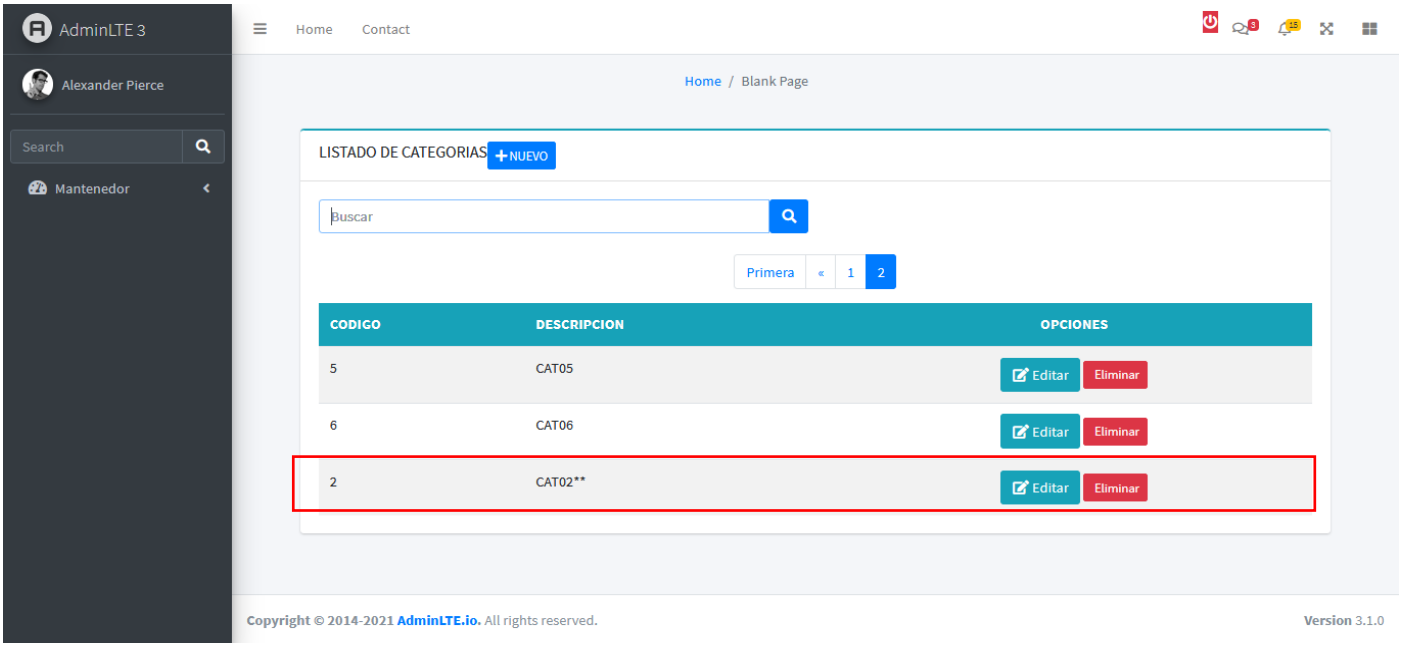

Figura 6: CAT02 fue editada

4. Eliminar

- En view.py colocamos:

```
def eliminarcategoriaMia(request,id):
    Categoria.objects.filter(id=id).update(estado=False)
    return redirect("listarcategoriaMia")
```

- Se actualiza el estado de la Categoría utilizando el ORM de Django:

```
Categoria.objects.filter(id=id).update(estado=False)
```

Obtiene la Categoría con tal id, y después actualizar el campo estado de tal Categoría.

- En urls.py:

```
path('eliminarcategoriaMia/<int:id>/',eliminarcategoriaMia ,name="elimina
rcategoriaMia"),
```

- Capturas sobre eliminación de categoría:

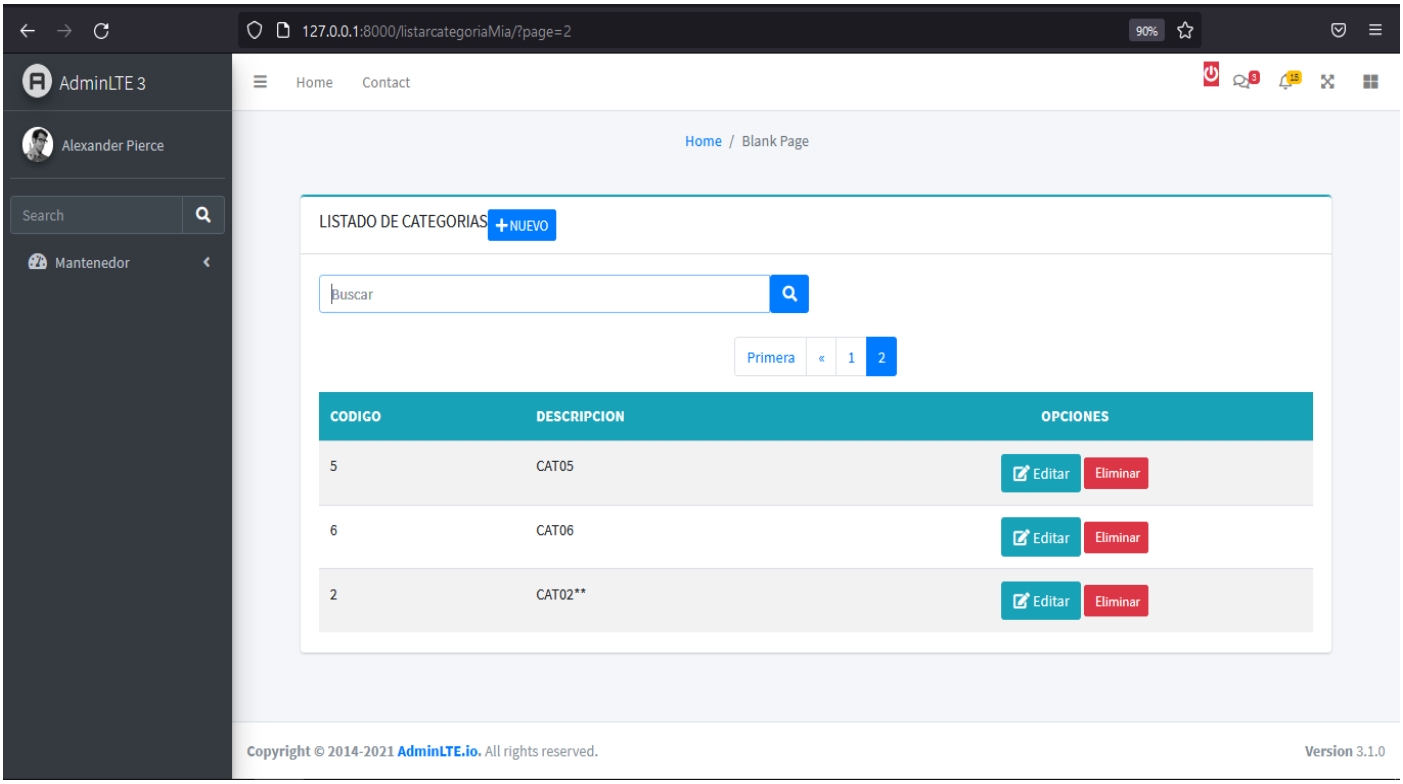

Figura 7: Eliminamos CAT06

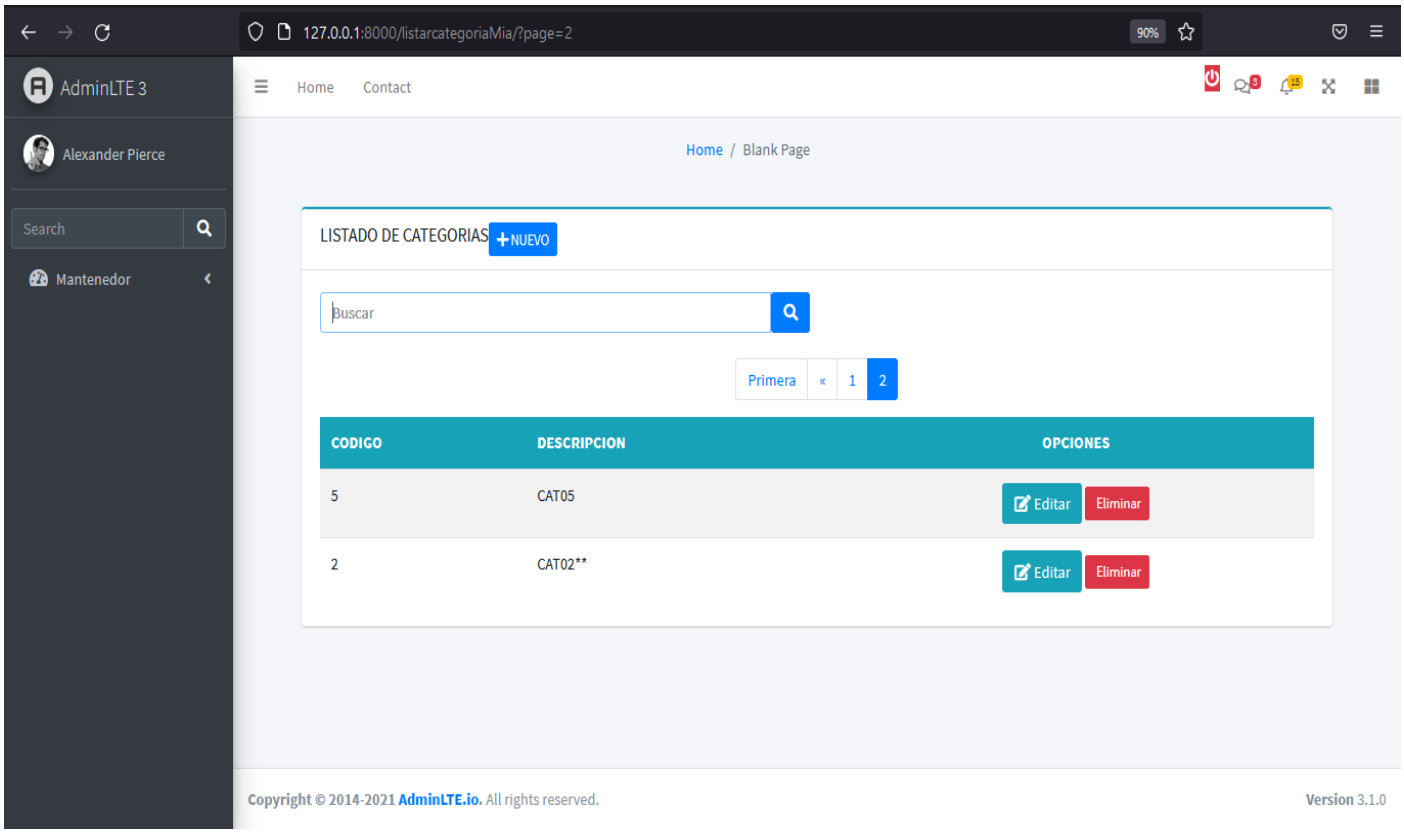

Figura 8: CAT06 eliminada

Referencias Bibliográficas

02dc5d3615b13698455a1e0ec07fd07d86557975 @ axiacore.com. (n.d.). https://axiacore.com/blog/paginacion-en-django-estilo-digg-409/

0eea7c4eabf4abf6d3707cc5f7a4cbbb0c70e514 @ unipython.com. (n.d.). https://unipython.com/concatenacion-de-strings-y-formato-en-python/

62c6e9de1d754ef03d5ed3d97af10ecf17b35d1a @ web.archive.org. (n.d.). https://web.archive.org/web/20110513122309/http://djangoadvent.com/1.2/smoothing-curve/

68804b3ee7188d0e2f7d3a75a70ed731625f2051 @ docs.djangoproject.com. (n.d.). https://docs.djangoproject.com/en/3.2/topics/db/sql/

736ec334840151543538eafd1d48fc00ad7a1a6e @ docs.djangoproject.com. (n.d.). https://docs.djangoproject.com/en/3.2/ref/request-response/

Models @ developer.mozilla.org. (n.d.). https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Models