

DUNIVERSIDAD TÉCNICA DE COTOPAXI

NOMBRE: RIVALDO GUTIÉRREZ

CURSO: 7MO "SISTEMAS"

FECHA: 4/1/2024

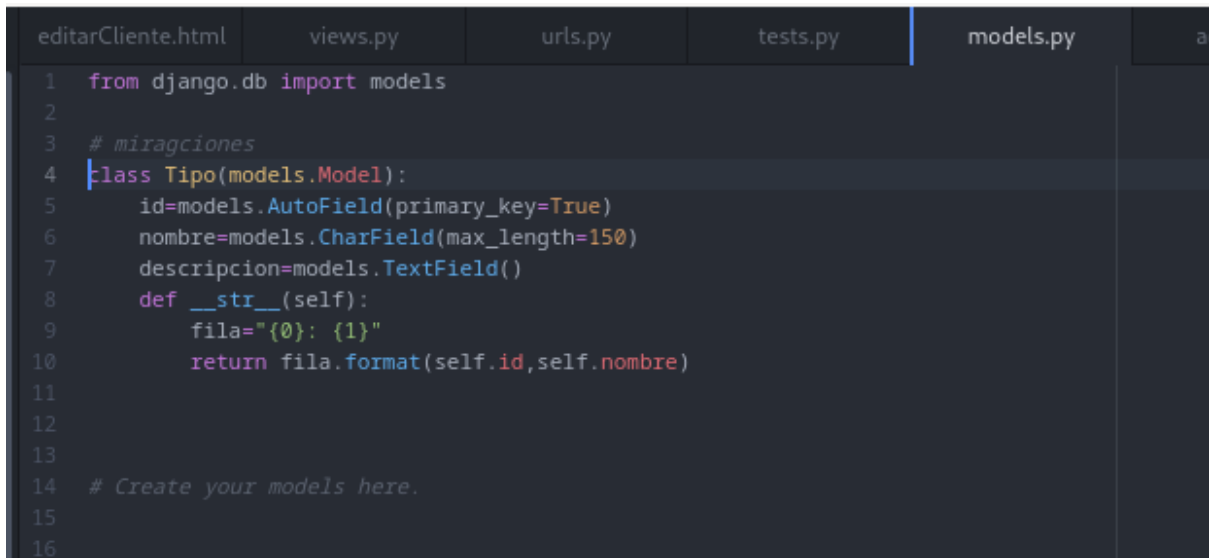
MANUAL DE INSERCIÓN CON RELACIONES ENTRE OBJETOS

PASO1: APAGAMOS EL SERVIDOR

Antes de realizar cualquier cambio, es crucial apagar el servidor para evitar conflictos durante la manipulación de datos.

PASO 2: CREAMOS EL MODELO TIPO

En este paso, procedemos a la creación del modelo "Tipo". Este modelo actuará como una entidad relacionada con otros objetos en nuestra aplicación.



```
1 from django.db import models
2
3 # miragciones
4 class Tipo(models.Model):
5     id=models.AutoField(primary_key=True)
6     nombre=models.CharField(max_length=150)
7     descripcion=models.TextField()
8     def __str__(self):
9         fila="{0}: {1}"
10        return fila.format(self.id,self.nombre)
11
12
13
14 # Create your models here.
15
16
```

PASO 3: CREAMOS EL ATRIBUTO TIPO, LLAMAMOS A LA CLASE CREADA, POR EL MÉTODO DE CASCADA

Al crear el atributo "Tipo" en otro modelo, se vincula a la clase creada mediante el método de cascada. Esto establecerá la relación entre los objetos.

```

class Cliente(models.Model):
    id=models.AutoField(primary_key=True)
    cedula=models.CharField(max_length=10)
    apellido=models.CharField(max_length=150)
    nombre=models.CharField(max_length=150)
    direccion=models.TextField()
    fecha_nacimiento=models.DateField()
    correo=models.EmailField()
    #llamamos a la clase Tipo, MEDIANTE CASACADA
    tipo=models.ForeignKey(Tipo, null=True,blank=True,on_delete=models.PROTECT)

    def __str__(self):

        fila="{0}: {1} {2} - {3}"
        return fila.format(self.cedula, self.apellido, self.nombre, self.correo)

```

PASO 4: EJECUTAMOS EL MAKEMIGRATIONS

Este paso asegura que los cambios realizados en los modelos se reflejen en la base de datos. Ejecutamos el comando "makemigrations" para generar las migraciones necesarias.

```

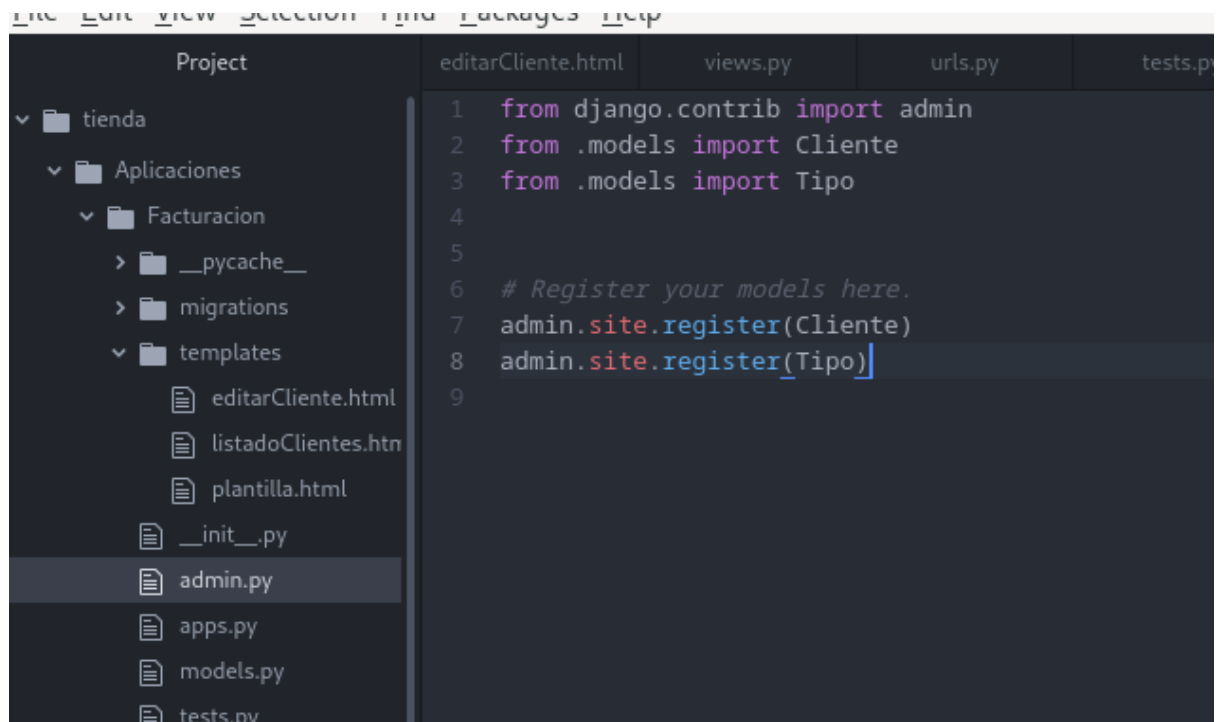
[rivaldo@fedora tienda]$ python manage.py makemigrations
Migrations for 'Facturacion':
  Aplicaciones/Facturacion/migrations/0002_tipo_cliente_tipo.py
    - Create model Tipo
    - Add field tipo to cliente
[rivaldo@fedora tienda]$ python manage.py migrate
Operations to perform:
  Apply all migrations: Facturacion, admin, auth, contenttypes, sessions
Running migrations:
  Applying Facturacion.0002_tipo_cliente_tipo... OK
[rivaldo@fedora tienda]$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
January 04, 2024 - 15:01:40
Django version 4.2.7, using settings 'tienda.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

```

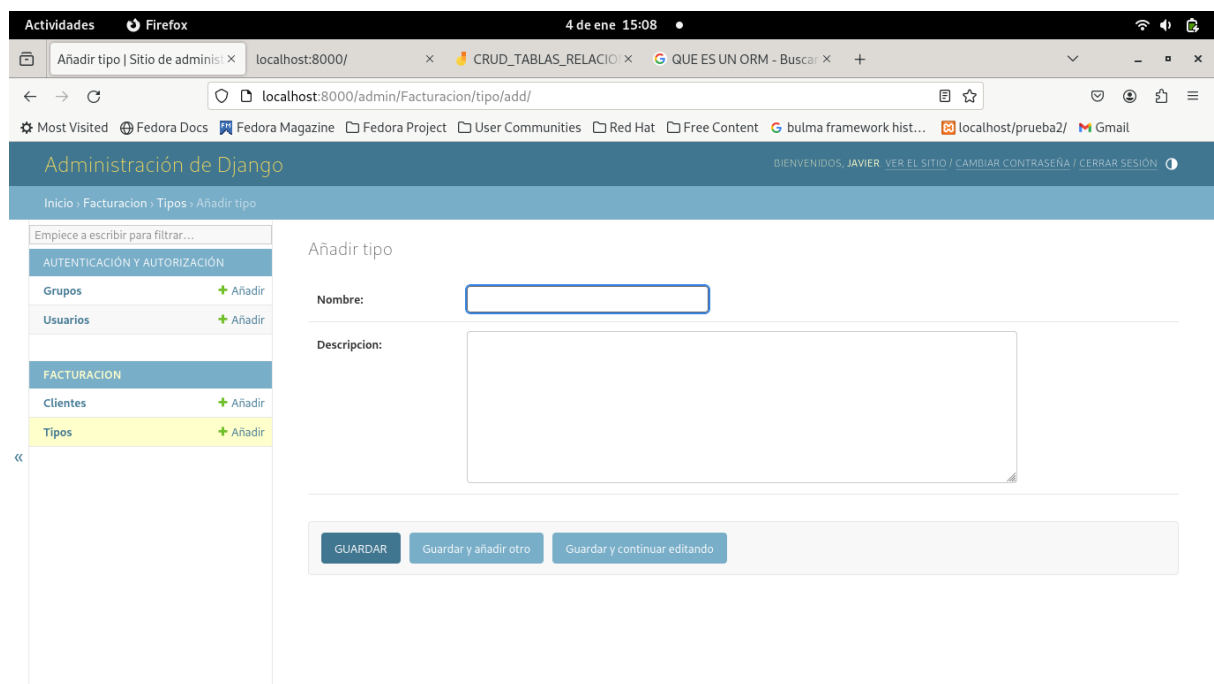
PASO 5: CREAMOS LA REDIRECCION PARA LA VISTA EN ADMIN.PY

Configuramos la redirección en el archivo "admin.py" para que la interfaz de administrador de Django muestre correctamente las relaciones entre objetos.



PASO 6: VERIFICAMOS QUE TODO FUNCIONE EN EL ADMINISTRADOR DE DJANGO

Antes de avanzar, es vital verificar que las relaciones entre objetos se hayan establecido correctamente en el administrador de Django.



PASO 7: VERIFICACIÓN

+ Añadir

Fecha nacimiento:

Hoy | 

Correo:

Tipo:

1: frecuente

2: ocasional

3: regular

GUARDAR

Guardar

Guardar y continuar editando

PASO 8: EDITAMOS EL FORMULARIO PARA AGREGAR EL NUEVO DATO

Modificamos el formulario correspondiente para incluir el nuevo dato "Tipo" y permitir su selección durante la creación o edición de un objeto.

```

d Packages Help
editarCliente.html views.py urls.py tests.py models.py admin.py listadoClientes... Telemetry Cons...

1  {% heredando header y footer%}
2  {% extends "../plantilla.html" %}
3  {% block contenido %}
4  {% load static %}
5
6      <h1>Listado de clientes</h1>
7  <div class="row">
8      <div class="col-md-5">
9          <form class="3" action="/guardarCliente/" method="post">
10             {% csrf_token %}
11
12             <select class="form-control" name="id_tipo" id="id_tipo" required>
13                 <option value="">Seleccione el tipo de cliente</option>
14             </select>
15
16             <input type="number" name="cedula" id="cedula" value=""placeholder="Ingrese la cedula" class="form-co
17             <br>
18             <input type="text" name="apellido" id="apellido" value=""placeholder="Ingrese el apellido" class="for
19
20             <br>
21             <input type="text" name="nombre" id="nombre" value=""placeholder="Ingrese el nombre" class="form-cont
22
23             <br>

```

PASO 9: AGREGAMOS LA LÓGICA DE NEGOCIO PARA QUE FUNCIONE NUESTRO NUEVO DATO

Implementamos la lógica necesaria en nuestras vistas para procesar y almacenar la información del nuevo dato "Tipo" correctamente.

```

d Packages Help
editarCliente.html views.py urls.py tests.py models.py admin.py listadoClientes... Telemetry Cons...

1  from django.shortcuts import render, redirect
2  from .models import Cliente
3  from django.contrib import messages
4  from datetime import datetime
5
6
7  def listadoClientes(request):
8      clientesBdd = Cliente.objects.all()
9      tiposBdd=Tipo.objects.all()
10     return render(request, 'listadoClientes.html', {'clientes': clientesBdd,'tipos':tiposBdd})
11
12
13
14  def guardarCliente(request):
15      cedula = request.POST["cedula"]
16      apellido = request.POST["apellido"]
17      nombre = request.POST["nombre"]
18      direccion = request.POST["direccion"]
19      fecha_nacimiento_str = request.POST["fecha_nacimiento"]
20      correo = request.POST["correo"]
21
22      # Formatear la fecha correctamente
23      try:

```

PASO 10: AGREGAMOS EL TEMPLATES PARA PODER SELECCIONAR LAS 3 OPCIONES

Creamos los templates necesarios que permitirán seleccionar entre las opciones disponibles para el nuevo dato "Tipo".

```
5
6 <h1>Listado de clientes</h1>
7 <div class="row">
8 <div class="col-md-5">
9 <form class="3" action="/guardarCliente/" method="post">
10 {% csrf_token %}
11
12 <select class="form-control" name="id_tipo" id="id_tipo" required>
13 <option value="">Seleccione el tipo de cliente</option>
14 {% for tipo in tipos %}
15 <option value="{{ tipo.id }}">{{ tipo.nombre }}</option>
16 {% endfor %}
17 </select>
18
19
20 <input type="number" name="cedula" id="cedula" value="" placeholder="Ingrese la cedula" class="form-co
21 <br>
22 <input type="text" name="apellido" id="apellido" value="" placeholder="Ingrese el apellido" class="fo
23
24 <br>
25 <input type="text" name="nombre" id="nombre" value="" placeholder="Ingrese el nombre" class="form-cont
26
27 <br>
28 <textarea name="direccion" id="direccion" rows="3" class="form-control" placeholder="Ingrese la direc
29 </textarea>
30 </div>
31 </div>
```

PASO 11: CAPTURAMOS AHORA, EL TIPO

En este paso, capturamos y procesamos la información relacionada con el nuevo dato "Tipo" durante la interacción con el formulario.

```
10 tiposBdd=Tipo.objects.all()
11 return render(request, 'listadoClientes.html', {'clientes': clientesBdd,'tipos':tiposBdd})
12
13
14
15 def guardarCliente(request):
16     #capturando los valores del formulario POST
17     id_tipo=request.POST["id_tipo"]
18     #capturando el tipo seleccionado por el usuario
19     tipoSeleccionado=Tipo.objects.get(id=id_tipo)
20     cedula = request.POST["cedula"]
21     apellido = request.POST["apellido"]
22     nombre = request.POST["nombre"]
23     direccion = request.POST["direccion"]
24     fecha_nacimiento_str = request.POST["fecha_nacimiento"]
25     correo = request.POST["correo"]
26
27     # Formatear la fecha correctamente
28     try:
29         fecha_nacimiento = datetime.strptime(fecha_nacimiento_str, "%Y-%m-%d").date()
30     except ValueError:
31         messages.error(request, 'Fecha de nacimiento inválida. Formato esperado: YYYY-MM-DD')
32         return redirect('/')
33
34     # Insertando datos mediante ORM de Django
35     nuevoCliente = Cliente.objects.create(
```

PASO12: INSERTAMOS EL TIPO EN LA ORM DE DJANGO

Añadimos el dato "Tipo" a la ORM (Object-Relational Mapping) de Django para que se refleje correctamente en la base de datos.

```
30     except ValueError:
31         messages.error(request, 'Fecha de nacimiento inválida. Formato esperado: YYYY-MM-DD')
32         return redirect('/')
33
34     # Insertando datos mediante ORM de Django
35     nuevoCliente = Cliente.objects.create(
36         cedula=cedula,
37         apellido=apellido,
38         nombre=nombre,
39         direccion=direccion,
40         fecha_nacimiento=fecha_nacimiento,
41         correo=correo,
42         tipo=tipoSeleccionado)
43
44     messages.success(request, 'Cliente guardado exitosamente')
45     return redirect('/')
46
47
48 def eliminarCliente(request, id):
49     clienteEliminar = Cliente.objects.get(id=id)
50     clienteEliminar.delete()
51     messages.error(request, 'Cliente eliminado exitosamente')
52     return redirect('/')
53
54
55 def editarCliente(request, id):
```

PASO 13: EN NUESTRO “ListadoClientes.html” , MOSTRAMOS EL APARTADO DEL NUEVO DATO

A ctualizamos la interfaz de "ListadoClientes.html" para mostrar la información relacionada con el nuevo dato "Tipo".

```

45     <br>
46     <table class="table">
47         <thead>
48             <tr>
49                 <th>ID</th>
50                 <th>CEDULA</th>
51                 <th>APELLIDO</th>
52                 <th>NOMBRE</th>
53                 <th>DIRECCION</th>
54                 <th>TIPO</th>
55                 <th>ACCIONES</th>
56             </tr>
57         </thead>
58         <tbody>
59             {% for cliente in clientes %}
60                 <tr>
61                     <td> {{cliente.id}}</td>
62                     <td> {{cliente.cedula}} </td>
63                     <td> {{cliente.apellido}}</td>
64                     <td>{{cliente.nombre}}</td>
65                     <td>{{cliente.direccion}}</td>
66                     <td>{{cliente.tipo}}</td>
67                     <td>
68                         <a href="{% url 'editarCliente' cliente.id %}" class="btn btn-warning">Editar</a>
69
70                         <a href="javascript:void(0)"onclick="eliminarCliente('/eliminarCliente/{{cliente.id}}');" class="

```

PASO 14: LO AJUSTAMOS PARA QUE SE PUEDA VISUALIZAR DE MEJOR MANERA

Realizamos ajustes en la presentación visual del "ListadoClientes.html" para mejorar la experiencia del usuario al visualizar el nuevo dato "Tipo".

```

45     <br>
46     <table class="table">
47         <thead>
48             <tr>
49                 <th>ID</th>
50                 <th>CEDULA</th>
51                 <th>APELLIDO</th>
52                 <th>NOMBRE</th>
53                 <th>DIRECCION</th>
54                 <th>TIPO</th>
55                 <th>ACCIONES</th>
56             </tr>
57         </thead>
58         <tbody>
59             {% for cliente in clientes %}
60                 <tr>
61                     <td> {{cliente.id}}</td>
62                     <td> {{cliente.cedula}} </td>
63                     <td> {{cliente.apellido}}</td>
64                     <td>{{cliente.nombre}}</td>
65                     <td>{{cliente.direccion}}</td>
66                     <td>{{cliente.tipo.nombre}}</td>
67                     <td>
68                         <a href="{% url 'editarCliente' cliente.id %}" class="btn btn-warning">Editar</a>
69
70                         <a href="javascript:void(0)"onclick="eliminarCliente('/eliminarCliente/{{cliente.id}}');" class="

```

PASO15: VERIFICAMOS

Antes de continuar, verificamos que la información relacionada con el nuevo dato "Tipo" se esté mostrando correctamente en la interfaz.

Actividades Firefox 4 de ene 15:37

localhost:8000/ localhost:8000

Captura de pantalla realizada
Puede pegar la imagen desde el portapapeles.

Tienda INICIO Entrada Dropdown Disabled Search Search

Listado de clientes

Seleccione el tipo de cliente

Ingrese la cedula


Ingrese el apellido

Ingrese el nombre

dd / mm / aaaa

Ingrese el correo

GUARDAR CANCELAR



ID	CEDULA	APELLIDO	NOMBRE	DIRECCION	TIPO	ACCIONES
32	0550639710	Gutierrez	Rivaldo	LATACUNGA	frecuente	Editar Eliminar
33	789	hhh	uu	ghj	frecuente	Editar Eliminar

PASO 16 : ACTUALIZACIÓN DEL BOTON EDITAR AÑADIENMDO EL NUEVO BOTON BOTÓN

En este paso, actualizamos la interfaz para incluir el nuevo botón que permitirá la edición del dato "Tipo" asociado a un cliente.

PASO 17: INGRESAMOS EL MISMO APARTADO DE ListadoClientes.html O LO COPIAMOS

Reutilizamos el apartado de "ListadoClientes.html" o copiamos su estructura para implementar la visualización y edición del nuevo dato "Tipo".

```

2 {% extends "../plantilla.html" %}
3 {% block contenido %}
4 {% load static %}
5
6 <h1>Listado de clientes</h1>
7 <div class="row">
8   <div class="col-md-5">
9     <form class="3" action="/guardarCliente/" method="post">
10       {% csrf_token %}
11
12       <select class="form-control" name="id_tipo" id="id_tipo" required>
13         <option value="">Seleccione el tipo de cliente</option>
14         {% for tipo in tipos %}
15           <option value="{{ tipo.id }}">{{ tipo.nombre }}</option>
16         {% endfor %}
17       </select>
18
19
20       <input type="number" name="cedula" id="cedula" value="" placeholder="Ingrese la cedula" class="form-co
21       <br>
22       <input type="text" name="apellido" id="apellido" value="" placeholder="Ingrese el apellido" class="foi
23
24       <br>
25       <input type="text" name="nombre" id="nombre" value="" placeholder="Ingrese el nombre" class="form-cont
26
27       <br>

```

PASO18: INGRESAMOS EL CODIGO EN EditarCliente.html O LO PEGAMOS

Integramos el código necesario en "EditarCliente.html" para permitir la edición del nuevo dato "Tipo" asociado a un cliente específico.

```

editarCliente.ht | views.py | urls.py | tests.py | models.py | admin.py | listadoClientes... | Telemetry Cons...
1 {# heredando header y footer #}
2 {% extends "../plantilla.html" %}
3 {% load static %}
4
5 {% block contenido %}
6 <div class="row">
7   <div class="col-md-2">
8   </div>
9   <div class="col-md-8">
10    <br> <br>
11    <form class="" action="{% url 'procesarActualizacionCliente' cliente.id %}" method="post">
12
13      {% csrf_token %}
14      <select class="form-control" name="id_tipo" id="id_tipo" required>
15        <option value="">Seleccione el tipo de cliente</option>
16        {% for tipo in tipos %}
17          <option value="{{ tipo.id }}">{{ tipo.nombre }}</option>
18        {% endfor %}
19      </select>
20
21
22
23      <input type="hidden" name="id" value="{{cliente.id}}">
24      <input type="text" name="" value="CEDULA">

```

PASO 19: AGREGAMOS LA FUNCIONALIDAD PARA QUE SE PUEDA EDITAR EL NUEVO DATO

Implementamos la lógica en las vistas correspondientes para permitir la edición del nuevo dato "Tipo" durante la modificación de un cliente.

```
editarCliente.html | views.py | urls.py | tests.py | models.py | admin.py | listadoClientes... | Telemetry Cons...
41 correo=correo,
42 tipo=tipoSeleccionado)
43
44 messages.success(request, 'Cliente guardado exitosamente')
45 return redirect('/')
46
47
48 def eliminarCliente(request, id):
49     clienteEliminar = Cliente.objects.get(id=id)
50     clienteEliminar.delete()
51     messages.error(request, 'Cliente eliminado exitosamente')
52     return redirect('/')
53
54
55 def editarCliente(request, id):
56     clienteEditar = Cliente.objects.get(id=id)
57     tiposBdd = Tipo.objects.all()
58     return render(request, 'editarCliente.html', {
59         'cliente': clienteEditar,
60         'tipos': tiposBdd
61     })
62
63
64
65 def procesarActualizacionCliente(request, id):
```

PASO 20: EN EDITAR CLIENTE AGREGAMOS UN SCRIPT PARA QUE SE NOS APAREZCA LOS DATOS DEL NUEVO DATO EN EL FORMULARIO

Añadimos un script JavaScript en la página "EditarCliente.html" para que la información del nuevo dato "Tipo" se muestre correctamente en el formulario de edición.

```
editarCliente.html | views.py | urls.py | tests.py | models.py | admin.py | listadoClientes... | Telemetry Con
1  {# heredando header y footer #}
2  {% extends "../plantilla.html" %}
3  {% load static %}
4
5  {% block contenido %}
6  <div class="row">
7      <div class="col-md-2">
8      </div>
9      <div class="col-md-8">
10         <br> <br>
11         <form class="" action="{% url 'procesarActualizacionCliente' cliente.id %}" method="post">
12
13             {% csrf_token %}
14             <select class="form-control" name="id_tipo" id="id_tipo" required>
15                 <option value="">Seleccione el tipo de cliente</option>
16                 {% for tipo in tipos %}
17                     <option value="{{ tipo.id }}">{{ tipo.nombre }}</option>
18                 {% endfor %}
19             </select>
20
21             <script type="text/javascript">
22                 document.getElementById("id_tipo")
23                     .value="{{ cliente.tipo.id }}";
24             </script>
25
26         </form>
27     </div>
28 </div>
29 </block contenido %}
```

PASO 21: AHORA CONTINUAMOS CON LA ACTUALIZACIÓN DEL NUEVO DATO

Proseguimos con la actualización del nuevo dato "Tipo" para garantizar que los cambios se reflejen adecuadamente en la base de datos.

```
64
65 def procesarActualizacionCliente(request, id):
66     id = request.POST["id"]
67     #agregamos estas lineas
68     id_tipo=request.POST["id_tipo"]
69     tipoSeleccionado=Tipo.objects.get(id=id_tipo)
70     #estas
71     cedula = request.POST["cedula"]
72     apellido = request.POST["apellido"]
73     nombre = request.POST["nombre"]
74     direccion = request.POST["direccion"]
75     fecha_nacimiento_str = request.POST["fecha_nacimiento"]
76     correo = request.POST["correo"]
77
78     # Formatear la fecha correctamente
79     try:
80         fecha_nacimiento = datetime.strptime(fecha_nacimiento_str, "%Y-%m-%d").date()
```

PASO 22: AGREGAMOS LA FUNCIONALIDAD PARA LA ACRUALIZACION DEL NUEVO DATO

Implementamos la lógica necesaria en las vistas para procesar y almacenar la actualización del nuevo dato "Tipo".

```
81 except ValueError:
82     messages.error(request, 'Fecha de nacimiento inválida. Formato esperado: YYYY-MM-DD')
83     return redirect('/')
84
85 # Insertando datos mediante ORM de Django
86 clienteEditar = Cliente.objects.get(id=id)
87 clienteEditar.tipo=tipoSeleccionado
88 clienteEditar.cedula = cedula
89 clienteEditar.apellido = apellido
90 clienteEditar.nombre = nombre
91 clienteEditar.direccion = direccion
92 clienteEditar.fecha_nacimiento = fecha_nacimiento
93 clienteEditar.correo = correo
94 clienteEditar.save()
95
96 messages.success(request, 'Cliente ACTUALIZADO Exitosamente')
97 return redirect('/')
```

PASO 23: VERIFICACION

Finalmente, verificamos que la actualización del nuevo dato "Tipo" se haya realizado correctamente y que la información se muestre adecuadamente en la interfaz.



ID	CEDULA	APELLIDO	NOMBRE	DIRECCION	TIPO	ACCIONES	
32	0550639710	Gutierrez	Rivaldo	LATACUNGA	ocasional	Editar	Eliminar
33	789	hhh	uu	ghj	frecuente	Editar	Eliminar

