# Manual técnico

Heli Saul, Vásquez Gómez, 201700852 Escuela de Mecánica Eléctrica, Facultad de Ingeniería, Universidad de San Carlos de Guatemala

Resumen: a continuación, se muestra los códigos necesarios para que alguien que tenga conocimientos en programación pueda optimizar el código.

B. Objetivos

1

Objetivo General

1. Mostrar los modelos creados

Objetivo especifico

1. Mostrar código fuente

#### C. comandos utilizados

```
from django.shortcuts import render,
                                               def store(request):
redirect
                                                   data = cartData(request)
from django.contrib import messages
                                                   cartItems = data['cartItems']
from django.contrib.auth.models import User
from django.contrib.auth import
authenticate, login
                                                   products = Product.objects.all()
from .models import *
from django.contrib.auth.forms import
                                               render(request, 'Generales/store.html', {"prod
UserCreationForm
                                               ucts": products ,'cartItems':cartItems})
from django.contrib.auth.forms import
AuthenticationForm
from .forms import UsuarioForm
from .utils import cookieCart , cartData
,guestOrder
import json
import datetime
                                               def cart(request):
from django.http import JsonResponse
# Create your views here.
                                                   data = cartData(request)
                                                   cartItems = data['cartItems']
                                                   order = data['order']
def register(request):
                                                   items = data['items']
    form = UserCreationForm()
```

l

if request.method == 'POST':

if form.is\_valid():

form.save()

'Generales/register.html', context)

# hacer algo después de guardar

UserCreationForm(request.POST)

context = {'form':form}

return render(request,

form =

los datos del usuario

orderItem.quantity =

```
(orderItem.quantity - 1)
    context = {'items': items,
                                                   orderItem.save()
'order':order,'cartItems': cartItems}
    return
                                                   if orderItem.quantity <= 0:</pre>
render(request, 'Generales/cart.html',context
                                                       orderItem.delete()
                                                   return JsonResponse('Item was
                                               added',safe=False)
def checkout(request):
    data = cartData(request)
    cartItems = data['cartItems']
                                               def processOrder(request):
    order = data['order']
                                                   transaction id =
    items = data['items']
                                               datetime.datetime.now().timestamp()
                                                   data = json.loads(request.body)
    context = {'items': items,
                                                   if request.user.is_authenticated:
'order':order,'cartItems':cartItems}
                                                       customer = request.user.customer
                                                       order, created =
    return
                                               Order.objects.get_or_create(customer=custome
render(request, 'Generales/checkout.html',con
                                               r,complete=False)
text)
                                                   else:
                                                       customer , order =
def updateItem(request):
                                               guestOrder(request,data)
    data = json.loads(request.body)
    productId = data['productId']
    action = data['action']
                                                   total = float(data['form']['total'])
    print('Action:',action)
                                                   order.transaction_id = transaction_id
    print('productId',productId)
                                                   if total == order.get_cart_total:
    customer = request.user.customer
                                                           order.complete = True
    product = Product.objects.get(id =
                                                   order.save()
productId)
                                                   if order.shipping == True:
    order, created =
                                                           ShippingAddress.objects.create(
Order.objects.get_or_create(customer=custome
                                                           customer = customer,
r,complete=False)
                                                           order = order,
    orderItem, created =
                                                           address =
OrderItem.objects.get_or_create(order =
                                               data['shipping']['address'],
order,product=product)
                                                           city = data['shipping']['city'],
                                                           state =
    if action == 'add':
                                               data['shipping']['state'],
        orderItem.quantity =
                                                           zipcode =
(orderItem.quantity + 1)
                                               data['shipping']['zipcode'],
    elif action == 'remove':
```

```
OPCIONES_ROL = [
                                                       ('usuario', 'Usuario normal'),
            )
                                                       ('admin', 'Administrador'),
    return JsonResponse('Pago completado
                                                   1
!',safe=False)
                                                   roles = models.CharField(max_length=7,
                                               choices=OPCIONES_ROL, default='usuario')
                                                   def __str__(self):
                                                       return f"{self.nombre}
                                               {self.apellidos} ({self.nombre_usuario})"
                                               class Product(models.Model):
                                                   name = models.CharField(max_length =
from django.db import models
                                               200, null = True)
from django.contrib.auth.models import User
                                                   price = models.FloatField()
# Create your models here.
                                                   digital = models.BooleanField(default =
                                               False, null = True)
class Customer(models.Model):
                                                   image = models.ImageField(null = True ,
    user =
                                               blank=True)
models.OneToOneField(User,on_delete =
                                                   def __str__(self):
models.CASCADE, null = True, blank = True)
                                                       return self.name
    name = models.CharField(max_length =
200, null=True)
                                                   @property
    email = models.CharField(max_length =
                                                   def imageUR1(self):
200, null = True)
                                                       try:
                                                           url = self.image.url
    def __str__(self):
                                                       except:
        return self.name
                                                           url = ''
                                                       return url
class Usuario(models.Model):
    # Campos de texto
                                               class Order(models.Model):
    nombre = models.CharField(max_length=50)
                                                   customer =
    apellidos =
                                               models.ForeignKey(Customer,on delete=
models.CharField(max_length=50)
                                               models.SET_NULL, null=True, blank=True)
    DPI = models.CharField(max length=50)
                                                   date_orderd =
    Telefono =
                                               models.DateTimeField(auto_now_add =True)
models.CharField(max_length=50)
                                                   complete = models.BooleanField(default =
    nombre_usuario =
                                               False, null = True, blank = False)
models.CharField(max_length=30, unique=True)
                                                   transaction_id =
    email = models.EmailField(unique=True)
                                               models.CharField(max_length = 200, null =
    contrasenia =
                                               True)
models.CharField(max length=50)
                                                   def __str__(self):
    con contrasenia =
                                                       return str(self.id)
models.CharField(max length=50)
```

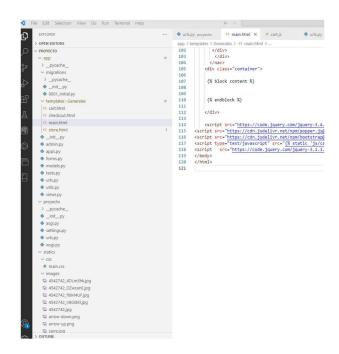
# Campos con opciones

```
@property
                                                          total = self.product.price *
    def shipping(self):
                                               self.quantity
        shipping = False
                                                          return total
        orderitems =
                                               class ShippingAddress(models.Model):
self.orderitem set.all()
        for i in orderitems:
                                                       customer =
            if i.product.digital == False:
                                               models.ForeignKey(Customer,on delete=
                shipping = True
                                               models.SET_NULL, null=True)
        return shipping
                                                       order =
                                               models.ForeignKey(Order,on_delete=
                                               models.SET_NULL, null=True)
                                                       address = models.CharField(max length
                                               = 200, null = False)
                                                       city = models.CharField(max length =
    @property
                                               200, null = False)
    def get_cart_total(self):
                                                       state = models.CharField(max_length =
                                               200, null = False)
        orderitems =
                                                       zipcode = models.CharField(max_length
self.orderitem_set.all()
                                               = 200, null = False)
        total = sum([item.get_total for item
                                                       date added =
in orderitems])
                                               models.DateTimeField(auto now add = True)
        return total
                                                       def __str__(self):
    @property
                                                        return str(self.address)
    def get cart items(self):
        orderitems =
self.orderitem set.all()
        total = sum([item.quantity for item
in orderitems])
        return total
                                               {% extends 'Generales/main.html' %}
                                               {% load static %}
                                               {% block content %}
class OrderItem(models.Model):
                                               <div class="row">
      product =
models.ForeignKey(Product,on delete=
models.SET NULL,null=True,blank=True)
                                                   {% for product in products %}
      order =
models.ForeignKey(Order,on delete=
                                                   <div class="col-lg-4">
models.SET_NULL,null=True,blank=True)
                                                        <img class="thumbnail"</pre>
      quantity = models.IntegerField(default
                                               src="{{product.imageUR1}}" >
= 0, null = True, blank = True)
                                                        <div class="box-element product">
      date added =
                                                            <h6><strong>{{product.name}}</st
models.DateTimeField(auto_now_add = True)
                                               rong></h6>
                                                            <hr>>
                                                            <button data-product</pre>
                                               ={{product.id}} data-action="add" class="btn
      @property
                                               btn-outline-secondary add-btn update-
      def get_total(self):
                                               cart">Asignarse al Curso</button>
```

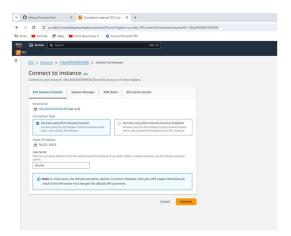
```
<a class="btn btn-outline-</pre>
                                                                   var cookies =
success" href="#">Ver Informacion del
                                                document.cookie.split(';');
curso</a>
                <h4 style="display: inline-
                                                                   for (var i = 0; i <</pre>
block; float:
                                                cookies.length; i++) {
right"><strong>Q{{product.price|floatformat:
2}}</strong></h4>
                                                                   var cookie =
                                                cookies[i].trim();
        </div>
    </div>
                                                                   // Does this cookie string
                                                begin with the name we want?
    {% endfor %}
                                                                   if (cookie.substring(0,
</div>
                                                name.length + 1) === (name + '=')) {
{% endblock content %}
                                                                   cookieValue =
                                                decodeURIComponent(cookie.substring(name.len
                                                gth + 1));
                                                                   break;
                                                                 }
                                                              }
<!DOCTYPE html>
{% load static %}
                                                      return cookieValue;
<html lang="en">
<head>
    <meta charset="UTF-8">
                                                        var csrftoken =
                                                getToken('csrftoken');
    <title>Document</title>
    <link rel="stylesheet"</pre>
                                                        function getCookie(name){
href="https://cdn.jsdelivr.net/npm/bootstrap
                                                          var cookieArr =
@4.4.1/dist/css/bootstrap.min.css"
                                                document.cookie.split(";");
integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiF
                                                          for(var i = 0; i <</pre>
eWPGFN9MuhOf23Q9Ifjh"
                                                cookieArr.length; i++){
crossorigin="anonymous">
  k
                                                                var cookiePair =
rel="stylesheet" type="text/css" href="{%
                                                cookieArr[i].split("=");
static 'css/main.css' %}">
                                                                if(name ==
    <meta name="viewport"</pre>
                                                cookiePair[0].trim()){
content="width=device-width, initial-scale=1
, maximum-scale = 1, minimum-scale = 1" />
                                                                  return
                                                decodeURIComponent(cookiePair[1]);
                                                                }
    <script type="text/javascript">
                                                          }
      var user ='{{request.user}}'
                                                         return null;
      function getToken(name) {
                                                        }
      var cookieValue = null;
                                                        var cart =
      if (document.cookie && document.cookie
!==''){
                                                JSON.parse(getCookie('cart'))
```

```
<a href="#"class="btn btn-</pre>
        if(cart == undefined){
                                              success">Registrarse</a>
             cart = {}
                                                           <a href="{% url 'cart' %}">
             console.log('La asignacion fue
                                                              <img id="cart-icon" src="{%</pre>
procesada')
                                              static 'images/cart.png' %}">
             document.cookie = 'cart='+
                                                          </a>
JSON.stringify(cart)+ ";domain=;path=/"
                                                          total">{{cartItems}}
        }
                                                      </div>
        console.log('Cart',cart)
                                                      </div>
                                                     </nav>
    </script>
                                                   <div class="container">
</head>
<body>
                                                   {% block content %}
    <nav class="navbar navbar-expand-lg</pre>
navbar-dark bg-dark">
                                                   {% endblock %}
        <a class="navbar-brand" href="{% url</pre>
'cart' %}">Academia USAC Ingenieria</a>
                                                  </div>
        <button class="navbar-toggler"</pre>
type="button" data-toggle="collapse" data-
                                                  <script
target="#navbarSupportedContent" aria-
                                               src="https://code.jquery.com/jquery-
controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle
                                               3.4.1.slim.min.js" integrity="sha384-
                                               J6qa4849b1E2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7im
navigation">
                                               GFAV0wwj1yYfoRSJoZ+n"
          <span class="navbar-toggler-</pre>
                                               crossorigin="anonymous"></script>
icon"></span>
                                               <script
        </button>
                                               src="https://cdn.jsdelivr.net/npm/popper.js@
                                               1.16.0/dist/umd/popper.min.js"
        <div class="collapse navbar-</pre>
                                               integrity="sha384-
collapse" id="navbarSupportedContent">
                                              Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzT
          tmI3UksdQRVvoxMfooAo"
            crossorigin="anonymous"></script>
              <a class="nav-link" href="{%</pre>
url 'cart' %}">Cursos <span class="sr-</pre>
                                               src="https://cdn.jsdelivr.net/npm/bootstrap@
only">(
                                              4.4.1/dist/js/bootstrap.min.js"
                current)
                                               integrity="sha384-
                                              wfSDF2E50Y2D1uUdj003uMBJnjuUD4Ih7YwaYd1iqfkt
              </span></a>
                                               j0Uod8GCEx130g8ifwB6"
            crossorigin="anonymous"></script>
                                               <script type="text/javascript" src="{%</pre>
          static 'js/cart.js' %}"></script>
          <div class="form-inline my-2 my-</pre>
                                               <script
                                                        src="https://code.jquery.com/jquer
lg-0">
                                                                integrity="sha256-
                                              y-3.1.1.min.js"
            <a href="#"class="btn btn-</pre>
                                              hVVnYaiADRTO2PzUGmuLJr8BLUSjGIZsDYGmIJLv2b8=
warning">Ingresar</a>
                                                  crossorigin="anonymous"></script>
            <hr>>
                                               </body>
```

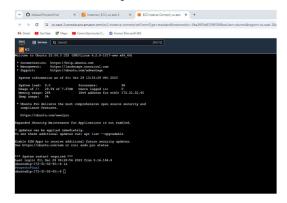
#### </html>



#### Mi Instancia



## Mi instancia en ejecucion



## conclusiones

## D.

- 1. Con este código cualquier otro que tenga el conocimiento en programas pueda optimizar el código presente
- 2. Se logro migrar todos los modelos de nuestro proyecto a la base de datos de postgres

## E. Código

 $\underline{https://github.com/helisaul/ProyectoFinal.git}$