[1]

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

Objetivo General

1. Mostrar los modelos creados

Objetivo especifico

1. Mostrar código fuente

C. comandos utilizados

```
from django.shortcuts import render,
redirect
from django.contrib import messages
from django.contrib.auth.models import User
from django.contrib.auth import
authenticate, login
from .models import *
from django.contrib.auth.forms import
UserCreationForm
from django.contrib.auth.forms import
AuthenticationForm
from .forms import UsuarioForm
from .utils import cookieCart , cartData
,guestOrder

import json
import datetime

from django.http import JsonResponse
# Create your views here.


def register(request):
    form = UserCreationForm()
    if request.method == 'POST':
        form =
UserCreationForm(request.POST)
        if form.is_valid():
            form.save()
            # hacer algo después de guardar
los datos del usuario
    context = {'form':form}
    return render(request,
'Generales/register.html', context)
```

```
def store(request):
    data = cartData(request)
    cartItems = data['cartItems']


    products = Product.objects.all()
    return
render(request,'Generales/store.html',{"prod
ucts": products ,'cartItems':cartItems})
```

```
def cart(request):

    data = cartData(request)
    cartItems = data['cartItems']
    order = data['order']
    items = data['items']
```

[1]

```
    context = {'items': items,
'order':order,'cartItems': cartItems}
    return
render(request,'Generales/cart.html',context
)

def checkout(request):

    data = cartData(request)
    cartItems = data['cartItems']
    order = data['order']
    items = data['items']

    context = {'items': items,
'order':order,'cartItems':cartItems}


    return
render(request,'Generales/checkout.html',con
text)



def updateItem(request):
    data = json.loads(request.body)
    productId = data['productId']
    action = data['action']
    print('Action:',action)
    print('productId',productId)

    customer = request.user.customer
    product = Product.objects.get(id =
productId)
    order, created =
Order.objects.get_or_create(customer=custome
r,complete=False)
    orderItem, created =
OrderItem.objects.get_or_create(order =
order,product=product)

    if action == 'add':
        orderItem.quantity =
(orderItem.quantity + 1)
    elif action == 'remove':
        orderItem.quantity =
(orderItem.quantity - 1)
    orderItem.save()

    if orderItem.quantity <= 0:
        orderItem.delete()
    return JsonResponse('Item was
added',safe=False)



def processOrder(request):
    transaction_id =
datetime.datetime.now().timestamp()
    data = json.loads(request.body)
    if request.user.is_authenticated:

        customer = request.user.customer
        order, created =
Order.objects.get_or_create(customer=custome
r,complete=False)



    else:

        customer , order =
guestOrder(request,data)


    total = float(data['form']['total'])
    order.transaction_id = transaction_id

    if total == order.get_cart_total:
            order.complete = True
    order.save()
    if order.shipping == True:
            ShippingAddress.objects.create(
            customer = customer,
            order = order,
            address =
data['shipping']['address'],
            city = data['shipping']['city'],
            state =
data['shipping']['state'],
            zipcode =
data['shipping']['zipcode'],
```

```
            )
    return JsonResponse('Pago completado
!',safe=False)
```

```
from django.db import models
from django.contrib.auth.models import User
# Create your models here.

class Customer(models.Model):
    user =
models.OneToOneField(User,on_delete =
models.CASCADE, null = True, blank = True)
    name = models.CharField(max_length =
200, null=True)
    email = models.CharField(max_length =
200, null = True)


    def __str__(self):
        return self.name


class Usuario(models.Model):
    # Campos de texto

    nombre = models.CharField(max_length=50)
    apellidos =
models.CharField(max_length=50)
    DPI = models.CharField(max_length=50)
    Telefono =
models.CharField(max_length=50)
    nombre_usuario =
models.CharField(max_length=30, unique=True)
    email = models.EmailField(unique=True)
    contrasenia =
models.CharField(max_length=50)
    con_contrasenia =
models.CharField(max_length=50)
    # Campos con opciones
    OPCIONES_ROL = [
        ('usuario', 'Usuario normal'),
        ('admin', 'Administrador'),
    ]
    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 =
200, null = True)
    price = models.FloatField()
    digital = models.BooleanField(default =
False, null = True)
    image = models.ImageField(null = True ,
blank=True)
    def __str__(self):
        return self.name

    @property
    def imageURl(self):
        try:

            url = self.image.url
        except:

            url = ''
        return url

class Order(models.Model):

    customer =
models.ForeignKey(Customer,on_delete=
models.SET_NULL,null=True,blank=True)
    date_orderd =
models.DateTimeField(auto_now_add =True)
    complete = models.BooleanField(default =
False, null = True, blank = False)
    transaction_id =
models.CharField(max_length = 200, null =
True)
    def __str__(self):
        return str(self.id)
```

```
    @property
    def shipping(self):
        shipping = False
        orderitems =
self.orderitem_set.all()
        for i in orderitems:
            if i.product.digital == False:
                shipping = True
        return shipping




    @property
    def get_cart_total(self):

        orderitems =
self.orderitem_set.all()
        total = sum([item.get_total for item
in orderitems])
        return total

    @property
    def get_cart_items(self):

        orderitems =
self.orderitem_set.all()
        total = sum([item.quantity for item
in orderitems])
        return total


class OrderItem(models.Model):
      product =
models.ForeignKey(Product,on_delete=
models.SET_NULL,null=True,blank=True)
      order =
models.ForeignKey(Order,on_delete=
models.SET_NULL,null=True,blank=True)
      quantity = models.IntegerField(default
= 0,null = True, blank = True)
      date_added =
models.DateTimeField(auto_now_add = True)



      @property
      def get_total(self):
          total = self.product.price *
self.quantity
          return total

class ShippingAddress(models.Model):
       customer =
models.ForeignKey(Customer,on_delete=
models.SET_NULL,null=True)
       order =
models.ForeignKey(Order,on_delete=
models.SET_NULL,null=True)
       address = models.CharField(max_length
= 200, null = False)
       city = models.CharField(max_length =
200, null = False)
       state = models.CharField(max_length =
200, null = False)
       zipcode = models.CharField(max_length
= 200, null = False)
       date_added =
models.DateTimeField(auto_now_add = True)

       def __str__(self):
        return str(self.address)
```

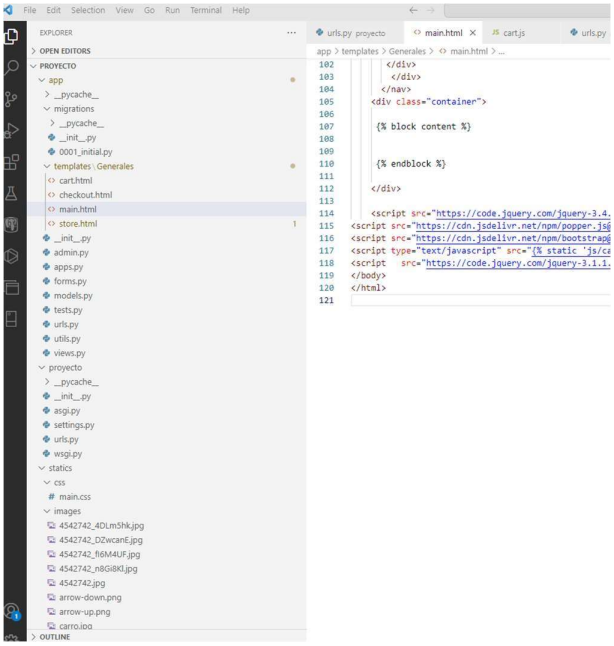
```
{% extends 'Generales/main.html' %}
{% load static %}
{% block content %}
<div class="row">


    {% for product in products %}

    <div class="col-lg-4">
        <img class="thumbnail"
src="{{product.imageURl}}" >
        <div class="box-element product">
            <h6><strong>{{product.name}}</st
rong></h6>
            <hr>
            <button data-product
={{product.id}} data-action="add" class="btn
btn-outline-secondary add-btn update-
cart">Asignarse al Curso</button>
```

```
            <a class="btn btn-outline-
success" href="#">Ver Informacion del
curso</a>
                <h4 style="display: inline-
block; float:
right"><strong>Q{{product.price|floatformat:
2}}</strong></h4>
        </div>
    </div>
    {% endfor %}
</div>
{% endblock content %}
```

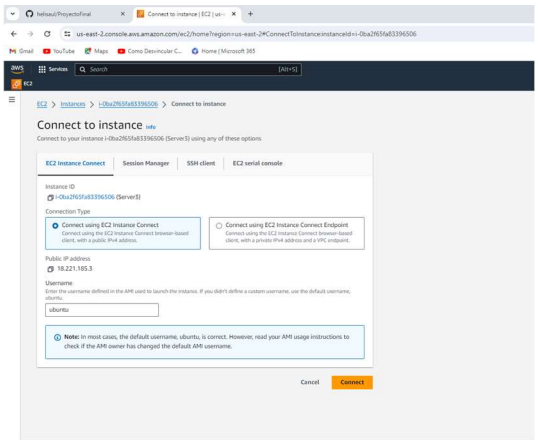
```
<!DOCTYPE html>
{% load static %}
<html lang="en">
<head>
    <meta charset="UTF-8">

    <title>Document</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap
@4.4.1/dist/css/bootstrap.min.css"
integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiF
eWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">
  <link
rel="stylesheet"  type="text/css"  href="{%
static 'css/main.css' %}">
    <meta name="viewport"
content="width=device-width, initial-scale=1
, maximum-scale = 1,minimum-scale = 1" />


    <script type="text/javascript">

      var  user ='{{request.user}}'


      function getToken(name) {
      var cookieValue = null;
      if (document.cookie && document.cookie
!==''){
                var cookies =
document.cookie.split(';');

                for (var i = 0; i <
cookies.length; i++) {

                var cookie =
cookies[i].trim();
                // Does this cookie string
begin with the name we want?
                if (cookie.substring(0,
name.length + 1) === (name + '=')) {
                cookieValue =
decodeURIComponent(cookie.substring(name.len
gth + 1));
                break;
                }
              }
            }
      return cookieValue;
      }

        var csrftoken =
getToken('csrftoken');

        function getCookie(name){
          var cookieArr =
document.cookie.split(";");

          for(var i = 0; i <
cookieArr.length; i++){

              var cookiePair =
cookieArr[i].split("=");
              if(name ==
cookiePair[0].trim()){

                return
decodeURIComponent(cookiePair[1]);

              }

          }

          return null;

        }
        var cart =
JSON.parse(getCookie('cart'))
```

```
        if(cart == undefined){

            cart = {}
            console.log('La asignacion fue
procesada')
            document.cookie = 'cart='+
JSON.stringify(cart)+ ";domain=;path=/"

        }

        console.log('Cart',cart)

    </script>
</head>
<body>
    <nav class="navbar navbar-expand-lg
navbar-dark bg-dark">
        <a class="navbar-brand" href="{% url
'cart' %}">Academia USAC Ingenieria</a>
        <button class="navbar-toggler"
type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle
navigation">
          <span class="navbar-toggler-
icon"></span>
        </button>

        <div class="collapse navbar-
collapse" id="navbarSupportedContent">
          <ul class="navbar-nav mr-auto">
            <li class="nav-item active">
              <a class="nav-link" href="{%
url 'cart' %}">Cursos <span class="sr-
only">(
                current)

              </span></a>
            </li>

          </ul>
          <div class="form-inline my-2 my-
lg-0">
            <a href="#"class="btn btn-
warning">Ingresar</a>
            <hr>
            <a href="#"class="btn btn-
success">Registrarse</a>

            <a href="{% url 'cart' %}">
               <img  id="cart-icon" src="{%
static 'images/cart.png' %}">
            </a>
            <p id="cart-
total">{{cartItems}}</p>

        </div>
        </div>
      </nav>
    <div class="container">

     {% block content %}


     {% endblock %}

    </div>

    <script
src="https://code.jquery.com/jquery-
3.4.1.slim.min.js" integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7im
GFAV0wwj1yYfoRSJoZ+n"
crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@
1.16.0/dist/umd/popper.min.js"
integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzT
tmI3UksdQRVvoxMfooAo"
crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@
4.4.1/dist/js/bootstrap.min.js"
integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfkt
j0Uod8GCExl3Og8ifwB6"
crossorigin="anonymous"></script>
<script type="text/javascript" src="{%
static 'js/cart.js' %}"></script>
<script   src="https://code.jquery.com/jquer
y-3.1.1.min.js"   integrity="sha256-
hVVnYaiADRTO2PzUGmuLJr8BLUSjGIZsDYGmIJLv2b8=
"   crossorigin="anonymous"></script>
</body>
```

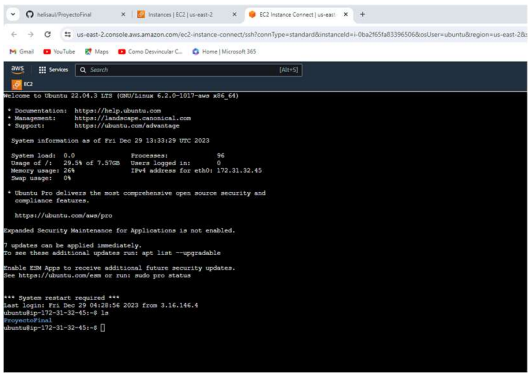
```
</html>
```



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*

https://github.com/helisaul/ProyectoFinal.git