

Universidad de San Carlos  
Facultad de Ingeniería  
Proyectos de computación aplicados a Ingeniería Electrónica

## Segundo Parcial

Autores:

**Henry José Chacón Barillas**  
**202002535**

**Andres Fernando Coronado Luna**  
**202006663**

Octubre, 2024

*El segundo parcial consiste en realizar una página de ventas utilizando Django*

### C. Bootstrap

Bootstrap es un framework de front-end gratuito y de código abierto que facilita el diseño y desarrollo de sitios web y aplicaciones web. Fue creado por desarrolladores de Twitter y es muy popular porque proporciona una serie de herramientas y componentes preconstruidos para crear interfaces web modernas, responsivas y estéticamente agradables sin necesidad de escribir mucho código CSS o JavaScript desde cero.

### I. OBJETIVOS

- Realizar el programa.
- Lograr diseñar la página requerida.
- Crear múltiples pestañas que permitan visualizar los productos, artículos del carrito y la entrega final.

### II. MARCO TEÓRICO

#### A. Django

Django es un framework web de alto nivel escrito en Python que permite desarrollar aplicaciones web de manera rápida y eficiente. Está diseñado para facilitar el desarrollo de sitios web complejos, ofreciendo una arquitectura robusta y reutilizable, con un enfoque en automatizar las tareas comunes y proporcionar herramientas útiles para los desarrolladores.



Figura 1: Django

#### B. Python

Python es un lenguaje de programación de alto nivel, interpretado y de propósito general. Es conocido por su legibilidad y facilidad de uso, lo que lo hace accesible tanto para principiantes como para programadores experimentados.



Figura 2: Python



Figura 3: Bootstrap

### III. ALGORITMO

La aplicación consiste en una venta de ropa en línea que no posee tienda física y realiza ventas en la modalidad de pago contra entrega.

Al entrar a la página se le solicita al cliente, su usuario y contraseña, si no tiene, se le deja crearlos, al entrar a la página principal se selecciona la cantidad de productos deseados para agregarlos al carrito.

Cuando el usuario termina y presiona el botón del carrito, se despliegan los productos que ha agregado, mostrando su total a pagar y pidiendo datos para su envío.

Al terminar se le enviará la confirmación del pedido al email del usuario y se almacenará su pedido para posteriormente imprimirla y entregar la información a los encargados de bodega y al repartidor para coordinar la entrega.

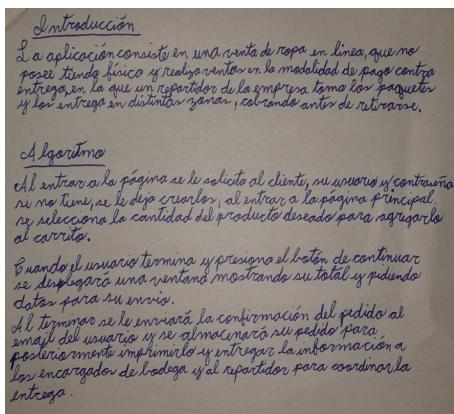
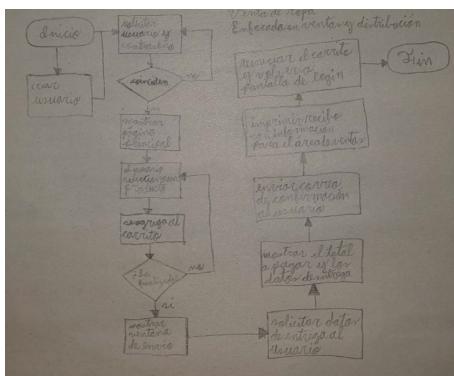
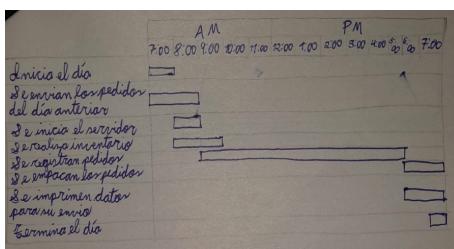


Figura 4: Algoritmo realizado en clase

## IV. ALGORITMO



## V. DIAGRAMA DE GANTT



## VI. CÓDIGOS

### A. Código principal de la página

```

1 <!DOCTYPE html>
2 {% load static %}
3
4
5 {% if messages %}
6   {% for message in messages %}
7     <div class="alert alert-info">{{ message
8   }}</div>
9   {% endfor %}
  
```

```

9   {% endif %}
10  <html>
11    <head>
12      <title>Proyecto Django</title>
13      <link href="https://cdn.jsdelivr.net/npm
14      /bootstrap@5.3.3/dist/css/bootstrap.min.css"
15      rel="stylesheet" integrity="sha384-
16      QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY
17      +ALEwIH" crossorigin="anonymous">
18      <link rel="stylesheet" type="text/css"
19      href="{% static 'css/main.css' %}">
20      <meta name="viewport" content="width=
21      device-width, initial-scale=1, maximum-scale
22      =1, minimum-scale=1" />
23      <style>
24          body {
25              background-image: url("{% static
26              'images/fondo.png' %}");
27              background-size: cover;
28              background-repeat: no-repeat;
29              background-attachment: fixed;
30              font-family: Helvetica, Arial,
31              sans-serif; /* Cambia la fuente a Helvetica
32              */
33          }
34          .navbar {
35              background-color: white !
36              important;
37          }
38          .navbar .navbar-brand,
39          .navbar .nav-link {
40              color: #151615 !important;
41          }
42          .navbar-brand-center {
43              position: absolute;
44              left: 50%;
45              transform: translateX(-50%);
46          }
47          .btn-login {
48              background-color: #7F7F7F !
49              important;
50              color: white;
51          }
52          .cart-container {
53              display: inline-block;
54              margin-left: 10px;
55          }
56          .welcome-image {
57              width: 100%;
58              max-width: 1920px;
59              height: auto;
60              display: block;
61              margin: 0 auto;
62          }
63          .separator {
64              width: 100%;
65              height: 2px;
66              background-color: #151615; /*
67              Lnea separadora de color #151615 */
68              margin: 20px 0;
69          }
70      </style>
71  </head>
72  <body>
73      <nav class="navbar navbar-expand-lg">
74          <!-- Parte superior izquierda -->
75          <a class="navbar-brand" href="{% url
76          'Tienda' %}">Premium Clothing</a>
77          <button class="navbar-toggler" type=
78          "button" data-bs-toggle="collapse" data-bs-
  
```

```

target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle
navigation">
64     <span class="navbar-toggler-icon">
65         </span>
66     </button>
67
68     <!-- Parte superior central con logo
-->
69     <a class="navbar-brand navbar-brand-
center" href="{% url 'Tienda' %}">
70         
71     </a>
72
73     <div class="collapse navbar-collapse
justify-content-end" id="
74 navbarSupportedContent">
75
76         <a href="http://127.0.0.1:8000/
login" class="btn btn-login">Login</a>
77
78         <div class="cart-container">
79             <div id="cart-total" class="text-center cart-number">0</div>
80             <a href="{% url 'Carrito' %}"
81             >
82                 <img id="cart-icon" src=
83                 "{% static 'images/carrito2.png' %}" alt="Carrito de compras">
84             </a>
85         </div>
86     </div>
87
88     <!-- Lnea separadora -->
89     <div class="separator"></div>
90
91     <div class="container">
92         {% block content %}
93         {% endblock content %}
94     </div>
95
96
97
98     <div class="container">
99         
100    </div>
101    <br/>
102
103    <script crossorigin="anonymous"></script
>
104 </body>
105 </html>

```

Listing 1: main.html

## B. Código principal de la tienda

```

1  {% extends 'generales/main.html' %}
2  {% load static %}

```

```

4  {% block content %}
5
6      <div class="container">
7          
8      </div>
9      <br/>
10
11 <div class="row">
12     <div class="col-lg-4">
13         <div class="product" id="product1">
14             
15             <div class="box-element">
16                 <h6><strong>Camisa Tommy
Hilfiger</strong></h6>
17                 <hr>
18                 <div class="counter">
19                     <button class="btn btn-
outline-secondary" onclick="decrement('hil1
')">#9664;</button>
20                     <span id="item-count1">0</
span>
21                     <button class="btn btn-
outline-secondary" onclick="increment('hil1
')">#9654;</button>
22                 </div>
23                 <br>
24                 <button class="btn btn-
outline-secondary add-btn" onclick="addToCart('hil1
', 'item-count1')">Agregar al carrito</
button>
25                     <a class="btn btn-outline-
success" href="tommy1">Ver</a>
26                     <h4 style="display: inline-block
; float: right"><strong>Q150</strong></h4>
27                 </div>
28             </div>
29
30
31     <div class="col-lg-4">
32         <div class="product" id="product2">
33             
34             <div class="box-element">
35                 <h6><strong>Hoodie Tommy
Hilfiger</strong></h6>
36                 <hr>
37                 <div class="counter">
38                     <button class="btn btn-
outline-secondary" onclick="decrement('hil2
')">#9664;</button>
39                     <span id="item-count2">0</
span>
40                     <button class="btn btn-
outline-secondary" onclick="increment('hil2
')">#9654;</button>
41                 </div>
42                 <br>
43                 <button class="btn btn-
outline-secondary add-btn" onclick="addToCart('hil2
', 'item-count2')">Agregar al carrito</
button>
44                     <a class="btn btn-outline-
success" href="tommy2">Ver</a>
45                     <h4 style="display: inline-block
; float: right"><strong>Q200</strong></h4>
46                 </div>
47             </div>

```

```

48     </div>
49
50     <div class="col-lg-4">
51         <div class="product" id="product2">
52             
53             <div class="box-element">
54                 <h6><strong>Sudadero Tommy Hilfiger</strong></h6>
55                 <hr>
56                 <div class="counter">
57                     <button class="btn btn-outline-secondary" onclick="decrement('hil2')">#9664;</button>
58                     <span id="item-count2">0</span>
59                     <button class="btn btn-outline-secondary" onclick="increment('hil2')">#9654;</button>
60                 </div>
61                 <br>
62                 <button class="btn btn-outline-secondary add-btn" onclick="addToCart('hil2', 'item-count2')">Agregar al carrito</button>
63                 <a class="btn btn-outline-success" href="#">Ver</a>
64                 <h4 style="display: inline-block; float: right"><strong>Q210</strong></h4>
65             </div>
66         </div>
67     </div>
68
69
70     <div class="container">
71         
72     </div>
73     <br/>
74
75     <div class="container">
76         
77     </div>
78     <br/>
79
80     <script>
81         let counts = {
82             hil1: parseInt(localStorage.getItem('hil1')) || 0,
83             hil2: parseInt(localStorage.getItem('hil2')) || 0,
84             hil3: parseInt(localStorage.getItem('hil3')) || 0,
85         };
86         const min = 0;
87         const max = 10;
88
89         function increment(product) {
90             if (counts[product] < max) {
91                 counts[product]++;
92                 updateDisplay(product);
93             }
94         }
95
96         function decrement(product) {
97             if (counts[product] > min) {
98                 counts[product]--;
99             }
100
101         updateDisplay(product);
102     }
103
104     </script>
105
106     <function updateDisplay(product) {
107         document.getElementById("item-count" + product.slice(3)).innerText = counts[product];
108         localStorage.setItem(product, counts[product]); // Guarda el conteo en localStorage
109     }
110
111     <function addToCart(productCode, countId) {
112         const count = counts[productCode];
113         let codigo;
114
115         // Determinar el código del producto según el identificador de productCode
116         switch (productCode) {
117             case 'hil1':
118                 codigo = 101;
119                 break;
120             case 'hil2':
121                 codigo = 102;
122                 break;
123             case 'hil3':
124                 codigo = 103;
125                 break;
126             default:
127                 console.error('Producto no válido');
128         }
129
130         // Enviar el código y la cantidad al servidor para actualizar la tabla
131         fetch('/update_cart', {
132             method: 'POST',
133             headers: {
134                 'Content-Type': 'application/json',
135                 'X-CSRFToken': '{{ csrf_token }}'
136             },
137             body: JSON.stringify({ producto: codigo, cantidad: count }),
138         })
139         .then(response => response.json())
140         .then(data => {
141             console.log('Success:', data);
142             // Resetea el contador después de agregar al carrito
143             counts[productCode] = 0;
144             updateDisplay(productCode);
145         })
146         .catch((error) => {
147             console.error('Error:', error);
148         });
149     }
150
151     </script>
152
153     {% endblock content %}

```

Listing 2: store.html

### C. Código para mostrar un producto

```

2 {% extends 'generales/main.html' %} 
3 {% load static %} 
4 {% block content %} 
5 <div style="background-color: #161616; 
    background-size: cover; background-position: 
        center; min-height: 100vh; padding: 50px 0; 
    > 
6     <div style="text-align: center; margin-top: 
        50px;"> 
7          
8         <h2 style="color: white;">Camisa Tommy 
        Hilfiger</h2> 
9         <br> 
10        <a href="http://127.0.0.1:8000/" class=" 
        btn" style="background-color: #161616; color: 
            white; padding: 10px 20px; text-decoration: 
            none; border: none; border-radius: 5px;"> 
11        Volver</a> 
12        <br><br> 
13         
14    </div> 
15 </div> 
16 {% endblock content %}

```

Listing 3: cart.html

## VII. RÚBRICA

Aspecto a Evaluar Requisitos	Descripción	Ponderación Sugerida	Autoevaluación
Cantidad y Complejidad	Los requisitos están definidos de manera clara, concisa y completa, sin ambigüedades.	15%	5%
Trazabilidad	Los requisitos se pueden rastrear desde el inicio hasta la implementación final.	10%	7%
Facilidad	Los requisitos son realistas y alcanzables dentro de las limitaciones del proyecto.	5%	4%
Diseño			
Arquitectura	La arquitectura del software es sólida, escalable y mantenible.	15%	9%
Diseno de Interfaces	Las interfaces de usuario son intuitivas, fáciles de usar y cumplen con los estándares de diseño.	10%	8%
Diseno de Base de Datos	La base de datos está bien diseñada, optimizada y cumple con los requisitos del sistema.	5%	4%
Implementación			
Calidad del Código	El código es limpio, bien estructurado, comentado y sigue las buenas prácticas de programación.	15%	7%
Pruebas Unitarias	Se han realizado pruebas unitarias exhaustivas para garantizar la corrección de cada componente.	10%	6%
Integración	Los diferentes componentes del sistema se integran correctamente y funcionan como un todo.	5%	4%
Documentación			
Guía del Usuario	La guía del usuario es clara, concisa y cubre todas las funcionalidades del sistema.	5%	4%
Documentación Técnica	Se ha generado una documentación técnica completa y detallada del sistema.	5%	1%
Pruebas			
Pruebas de Integración	Se han realizado pruebas de integración para verificar que los componentes interactúan correctamente.	5%	4%
Pruebas de Sistema	Se han realizado pruebas de sistema para verificar que el sistema cumple con los requisitos funcionales y no funcionales.	5%	3%
Mantenibilidad			
Modularidad	El sistema está dividido en módulos bien definidos y fáciles de mantener.	5%	5%
Reusabilidad	El código es reutilizable y puede ser utilizado en otros proyectos.	5%	5%

Figura 5: Rúbrica al día viernes 11

Aspecto a Evaluar Requisitos	Descripción	Ponderación Sugerida	Autoevaluación
Cantidad y Complejidad	Los requisitos están definidos de manera clara, concisa y completa, sin ambigüedades.	15%	5%
Trazabilidad	Los requisitos se pueden rastrear desde el inicio hasta la implementación final.	10%	8%
Facilidad	Los requisitos son realistas y alcanzables dentro de las limitaciones del proyecto.	5%	4%
Diseño			
Arquitectura	La arquitectura del software es sólida, escalable y mantenible.	15%	12%
Diseno de Interfaces	Las interfaces de usuario son intuitivas, fáciles de usar y cumplen con los estándares de diseño.	10%	10%
Diseno de Base de Datos	La base de datos está bien diseñada, optimizada y cumple con los requisitos del sistema.	5%	2%
Implementación			
Calidad del Código	El código es limpio, bien estructurado, comentado y sigue las buenas prácticas de programación.	15%	14%
Pruebas Unitarias	Se han realizado pruebas unitarias exhaustivas para garantizar la corrección de cada componente.	10%	9%
Integración	Los diferentes componentes del sistema se integran correctamente y funcionan como un todo.	5%	4%
Documentación			
Guía del Usuario	La guía del usuario es clara, concisa y cubre todas las funcionalidades del sistema.	5%	4%
Documentación Técnica	Se ha generado una documentación técnica completa y detallada del sistema.	5%	5%
Pruebas			
Pruebas de Integración	Se han realizado pruebas de integración para verificar que los componentes interactúan correctamente.	5%	4%
Pruebas de Sistema	Se han realizado pruebas de sistema para verificar que el sistema cumple con los requisitos funcionales y no funcionales.	5%	3%
Mantenibilidad			
Modularidad	El sistema está dividido en módulos bien definidos y fáciles de mantener.	5%	5%
Reusabilidad	El código es reutilizable y puede ser utilizado en otros proyectos.	5%	5%

Figura 6: Rúbrica al día viernes 18

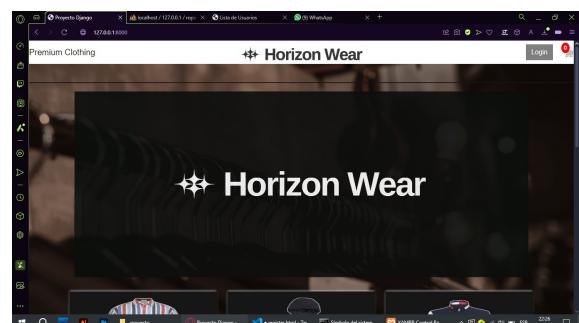
## VIII. MANUALES

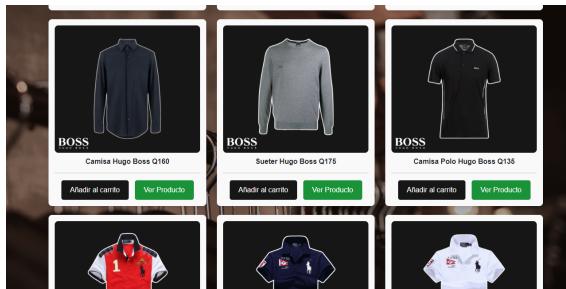


Figura 7: Manual de usuario

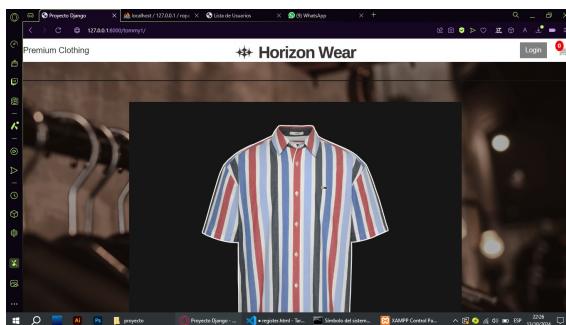
## IX. RESULTADOS

### A. Pantalla principal de la tienda

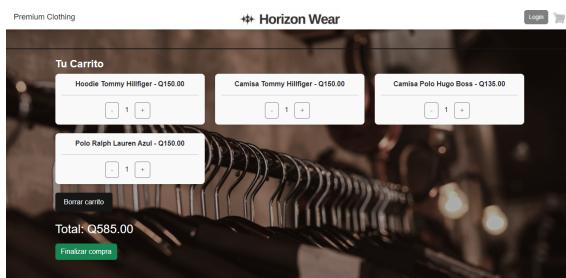




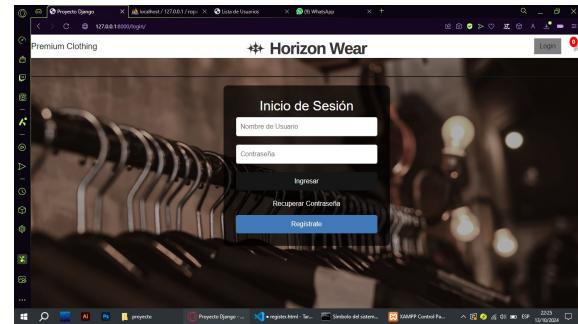
**B. Vista a un producto**



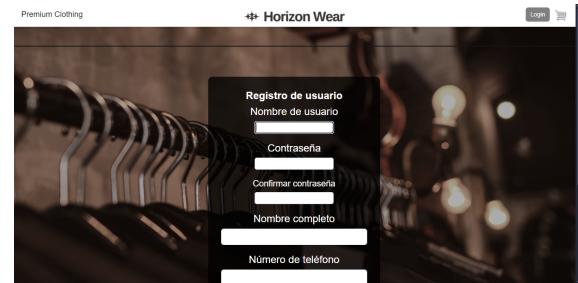
**C. Carrito**



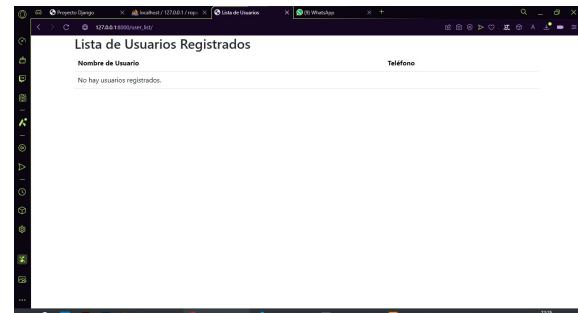
**D. Pantalla de inicio de sesión**



**E. Pantalla de registro**



**F. Pantalla de usuarios registrados**



**X. CONCLUSIONES**

- La modularidad del código y el uso de un framework como Django aseguran que el proyecto sea fácilmente escalable en el futuro. La estructura clara del código facilita su mantenimiento y la adición de nuevas funcionalidades.
- El uso de plantillas HTML y Bootstrap no solo permitió crear un diseño web atractivo, sino que también facilita futuras modificaciones y expansiones en la interfaz de usuario.

## XI. REPOSITORIO

<https://github.com/AndresCoronadoLuna/ProyectosDeComputacion>. [Fuente: Elaboración propia

2024].

[https://github.com/Henry-Chacon/Segundo\\_Parcial\\_Proyectos](https://github.com/Henry-Chacon/Segundo_Parcial_Proyectos). [Fuente: Elaboración propia 2024].