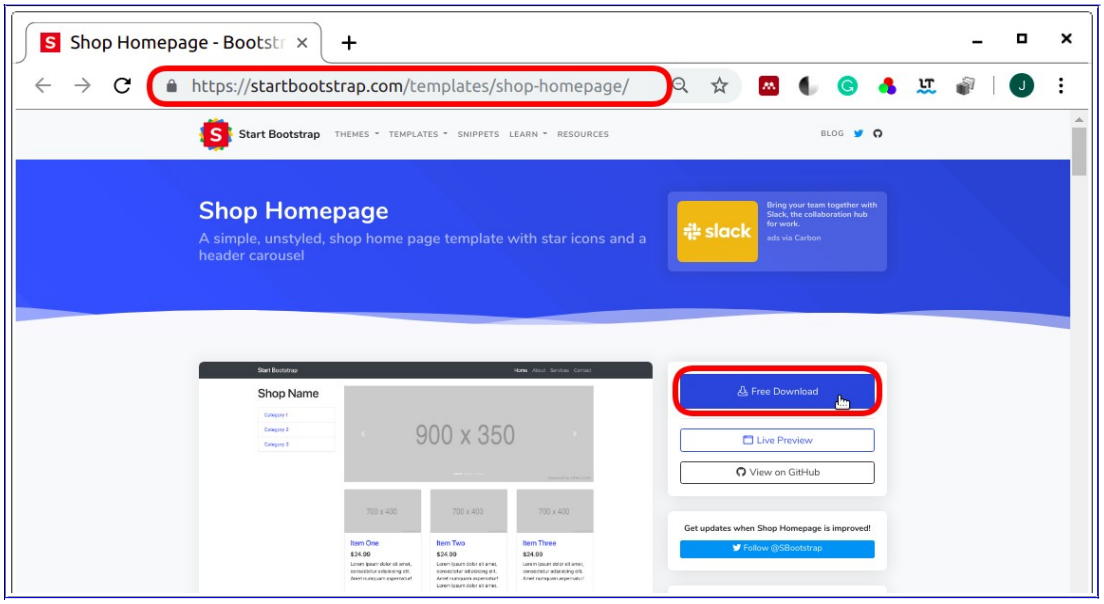
TIENDA DE COMERCIO ELECTRÓNICO

ITERACIÓN 05 – AJAX COMPONENTS

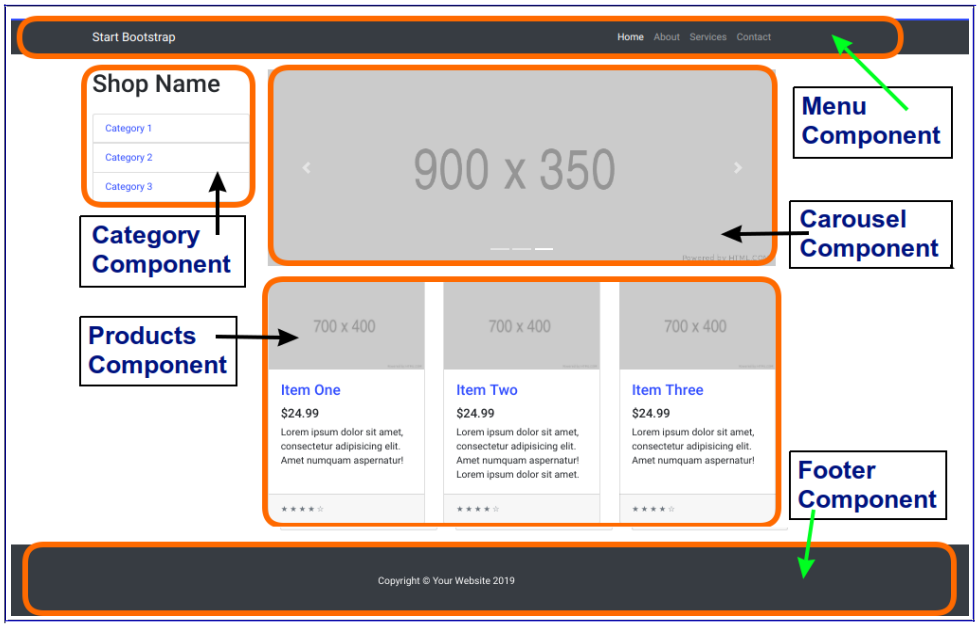
Autor: Mag. Juan Antonio Castro Silva
Versión: 2.0 Julio 28 de 2019 (20190728T2220)

1. IDENTIFICAR LOS COMPONENTES

Para esta aplicación se va utilizar una plantilla de bootstrap llamada shop-homepage, la cual se encuentra en la siguiente url: [https://startbootstrap.com/templates/shop-homepage/]. Para descargar esta plantilla haga click al lado derecho en el botón [Free Download].



Al analizar la página [shop-homepage] se pueden identificar los componentes web que la definen.



2. CREAR LOS SERVICIOS WEB

Archivo org.software.portal/PortalCategoryServervice.java:

```
001 package org.software.portal;
002
003 import java.sql.Connection;
004 import java.sql.ResultSet;
005 import java.sql.Statement;
006 import java.util.ArrayList;
007
008 import javax.ws.rs.GET;
009 import javax.ws.rs.Path;
010 import javax.ws.rs.Produces;
011
012 import org.software.category.Category;
013 import org.software.category.CategoryList;
014 import org.software.util.DataBase;
015
016 @Path("/portal")
017 public class PortalCategoryService {
018
019     @GET
020     @Path("/categories")
021     @Produces("application/json")
022     public CategoryList read() {
023         ArrayList<Category> categoryList = new ArrayList<Category>();
024         DataBase database = new DataBase();
025         Connection connection1 = null;
026         Statement statement1 = null;
027         ResultSet rs1 = null;
028         String sql = "";
029
030         try {
031             connection1 = database.getConnection("guest");
032             statement1 = connection1.createStatement();
033
034             sql = "select * from categories where published = 1";
035             sql += " order by name";
036
037             rs1 = statement1.executeQuery(sql);
038             while (rs1.next()) {
039                 int id = rs1.getInt("id");
040                 int published = rs1.getInt("published");
041                 String name = rs1.getString("name");
042                 String icon = rs1.getString("icon");
043
044                 Category category = new Category();
045                 category.setId(id);
046                 category.setPublished(published);
047                 category.setName(name);
048                 category.setIcon(icon);
049                 categoryList.add(category);
050             }
051         } catch (Exception e) {
052             System.out.println("Error: " + e.toString());
053         } finally {
054             database.closeObject(rs1);
055             database.closeObject(statement1);
056             database.closeObject(connection1);
```

```
057         }
058         return new CategoryList(categoryList);
059     }
060 }
```

Archivo org.software.product/Product.java:

```
001 package org.software.product;
002 
003 public class Product {
004 
005     long id;
006     int published;
007     int category_id;
008     String name;
009     double pricing;
010     String short_description;
011     String long_description;
012     String icon;
013 
014     public Product() {
015         super();
016     }
017 
018     public Product(long id, int published, int category_id, String name,
019         double pricing, String short_description,
020         String long_description, String icon) {
021 
022         super();
023         this.id = id;
024         this.published = published;
025         this.category_id = category_id;
026         this.name = name;
027         this.pricing = pricing;
028         this.short_description = short_description;
029         this.long_description = long_description;
030         this.icon = icon;
031     }
032 
033     public long getId() {
034         return id;
035     }
036 
037     public void setId(long id) {
038         this.id = id;
039     }
040 
041     public int getPublished() {
042         return published;
043     }
044 
045     public void setPublished(int published) {
046         this.published = published;
047     }
048 
049     public int getCategory_id() {
050         return category_id;
051     }
052 
053     public void setCategory_id(int category_id) {
054         this.category_id = category_id;
```

```
055     }
056
057     public String getName() {
058         return name;
059     }
060
061     public void setName(String name) {
062         this.name = name;
063     }
064
065     public double getPricing() {
066         return pricing;
067     }
068
069     public void setPricing(double pricing) {
070         this.pricing = pricing;
071     }
072
073     public String getShort_description() {
074         return short_description;
075     }
076
077     public void setShort_description(String short_description) {
078         this.short_description = short_description;
079     }
080
081     public String getLong_description() {
082         return long_description;
083     }
084
085     public void setLong_description(String long_description) {
086         this.long_description = long_description;
087     }
088
089     public String getIcon() {
090         return icon;
091     }
092
093     public void setIcon(String icon) {
094         this.icon = icon;
095     }
096 }
```

Archivo org.software.product/ProductList.java:

```
001 package org.software.product;
002
003 import java.util.List;
004
005 import javax.xml.bind.annotation.XmlElement;
006 import javax.xml.bind.annotation.XmlRootElement;
007
008 @XmlRootElement(name="products")
009 public class ProductList {
010
011     private List<Product> items;
012
013     public ProductList() {
014     }
015
016     public ProductList(List<Product> items) {
```

```
017         this.items = items;
018     }
019 
020     @XmlElement(name = "data")
021     public List<Product> getItems() {
022         return items;
023     }
024 }
```

Archivo org.software.portal/PortalProductServervice.java:

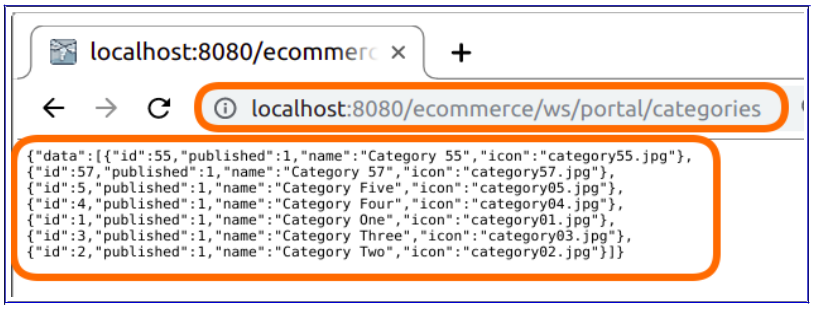
```
001 package org.software.portal;
002 
003 import java.sql.Connection;
004 import java.sql.ResultSet;
005 import java.sql.Statement;
006 import java.util.ArrayList;
007 
008 import javax.ws.rs.GET;
009 import javax.ws.rs.Path;
010 import javax.ws.rs.PathParam;
011 import javax.ws.rs.Produces;
012 
013 import org.software.product.Product;
014 import org.software.product.ProductList;
015 import org.software.util.DataBase;
016 
017 @Path("/portal")
018 public class PortalProductService {
019 
020     @GET
021     @Path("/products/{category}")
022     @Produces("application/json")
023     public ProductList getProducts(@PathParam(value = "category") int category) {
024         ArrayList<Product> productList = new ArrayList<Product>();
025         DataBase database = new DataBase();
026         Connection connection1 = null;
027         Statement statement1 = null;
028         ResultSet rs1 = null;
030         String sql = "";
031 
032         try {
033             connection1 = database.getConnection("guest");
034             statement1 = connection1.createStatement();
035 
036             sql = "select * from products where category_id = " + category;
037             sql += " and published = 1 order by name";
038 
039             rs1 = statement1.executeQuery(sql);
040             while (rs1.next()) {
041                 long id = rs1.getLong("id");
042                 int category_id = rs1.getInt("category_id");
040                 int published = rs1.getInt("published");
42                  String name = rs1.getString("name");
043                 double pricing = rs1.getDouble("pricing");
044                 String icon = rs1.getString("icon");
045                 String short_description = rs1.getString("short_description");
046                 //String long_description = rs1.getString("long_description");
047 
048                 Product product = new Product();
049                 product.setId(id);
```

```
029                 product.setCategory_id(category_id);
050                 product.setPublished(published);
051                 product.setName(name);
052                 product.setPricing(pricing);
050                 product.setIcon(icon);
52                  product.setShort_description(short_description);
053                 //product.setLong_description(long_description);
054                 productList.add(product);
055             }
056         } catch (Exception e) {
057             System.out.println("Error: " + e.toString());
058         } finally {
059             database.closeObject(rs1);
060             database.closeObject(statement1);
061             database.closeObject(connection1);
062         }
063         return new ProductList(productList);
064     }
    }
```

3. PROBAR LOS SERVICIOS WEB

3.1 Servicio Web Categories

En un navegador pruebe el método GET del servicio web categories: [http://localhost:8080/ecommerce/ws/portal/categories].



3.2 Servicio Web Products

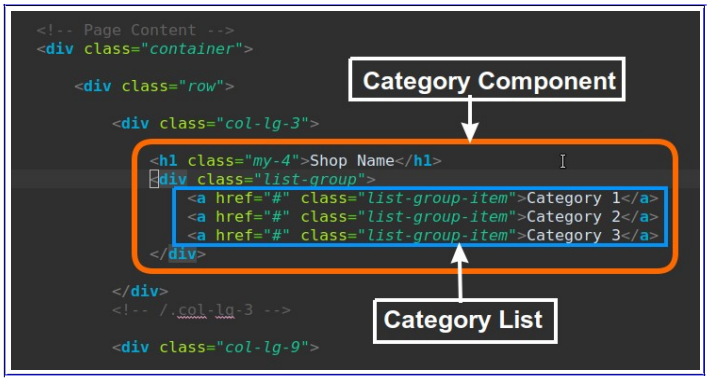
En un navegador pruebe el método GET del servicio web products: [http://localhost:8080/ecommerce/ws/portal/products/1].

4. CREAR LOS COMPONENTES WEB

En esta iteración se van a crear los componentes web Category y Products.

4.1 Category Component

En el código html se puede identificar el componente category y la lista de categorías.



Cuando se haga click en una categoría el sistema debe mostrar los productos pertenecientes a esta categoría. Para ello se implementará una función en JavaScript (jQuery) llamada getProducts(id) que cargará los productos en el componente Products.

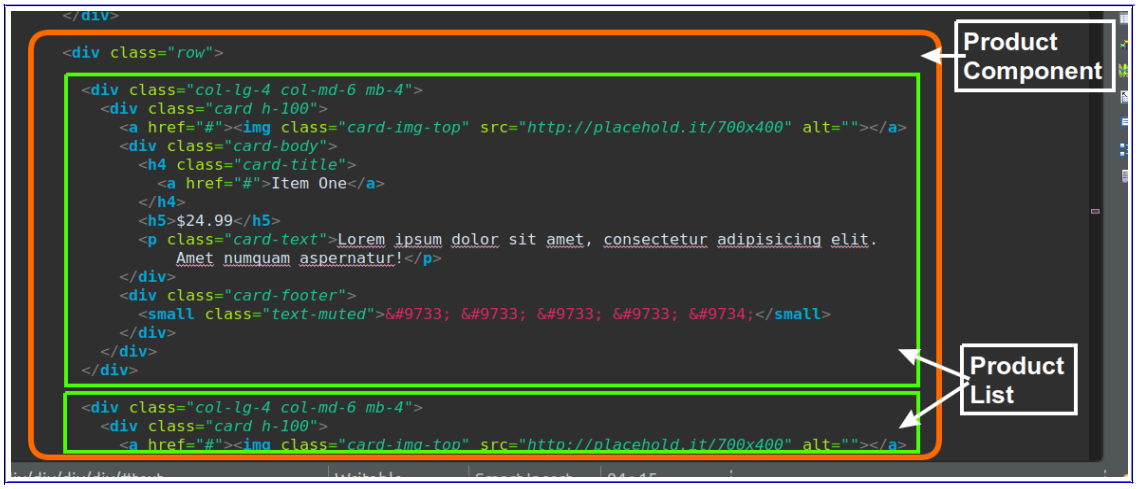
Cree el componente category en la carpeta portal.

Archivo portal/category.jsp:

```
001  <h1 class="my-4">Shop Name</h1>
002  <div class="list-group" id="div_categories">
003  </div>
```

4.2 Product Component

El código html muestra la sección (div) donde se agrupan los productos.

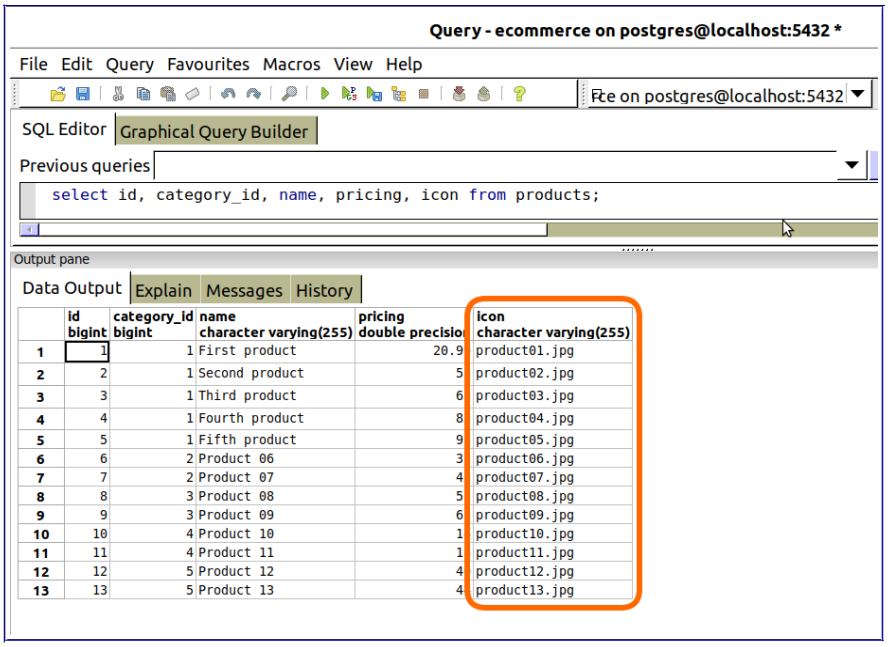


Cree el componente products en la carpeta portal.

Archivo portal/products.jsp:

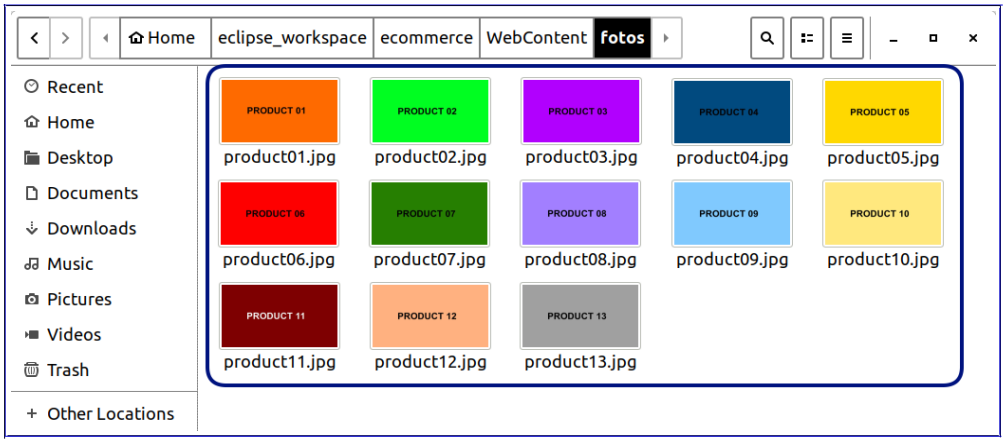
```
001  <div class="row" id="div_products">
002  </div>
003  <!-- /.row -->
```

El campo icon de la tabla products almacena la url de la imagen del producto.



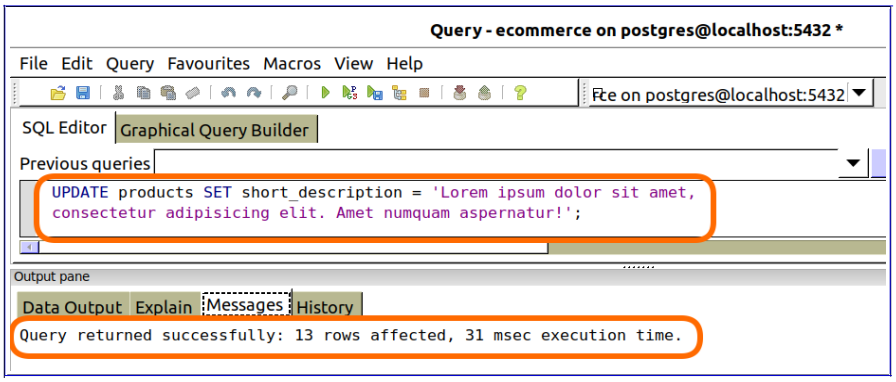
	id bigint	category_id bigint	name character varying(255)	pricing double precisio	icon character varying(255)
1	1	1	First product	20.9	product01.jpg
2	2	1	Second product	5	product02.jpg
3	3	1	Third product	6	product03.jpg
4	4	1	Fourth product	8	product04.jpg
5	5	1	Fifth product	9	product05.jpg
6	6	2	Product 06	3	product06.jpg
7	7	2	Product 07	4	product07.jpg
8	8	3	Product 08	5	product08.jpg
9	9	3	Product 09	6	product09.jpg
10	10	4	Product 10	1	product10.jpg
11	11	4	Product 11	1	product11.jpg
12	12	5	Product 12	4	product12.jpg
13	13	5	Product 13	4	product13.jpg

Se debe crear un folder llamado fotos en la carpeta [WebContent] para almacenar los archivos con las imágenes de los productos. Los nombres de los archivos deben corresponder con los valores registrados en el campo icon de la tabla products.



Actualice los valores del campo [short_description] de todos los registros de la tabla products.

```
001 UPDATE products SET short_description = 'Lorem ipsum dolor sit amet,
    consectetur adipisicing elit. Amet numquam aspernatur!';
```



5. LLAMADO A LOS SERVICIOS WEB

Para consumir los servicios web creados (categories y products), cree un archivo JavaScript llamado portal.js en la carpeta [WebContent/js].

Archivo js/portal.js:

```
001 function getCategories(category_id){
002     $.getJSON("../ws/portal/categories", function(result){
003         data = result.data;
004         $("#div_categories").empty();
005         for(var row=0; row<data.length; row=row+1){
006             var id = data[row].id;
007             var name = data[row].name;
008             var published = data[row].published;
009             var icon = data[row].icon;
010             var item_class = "list-group-item";
011             if(id == category_id){
012                 item_class = "list-group-item active";
013             }
014             $("#div_categories").append("<a href='javascript:getProducts(" + id
015                 + ");' id='category_" + id + "' class='" + item_class + "'>"
016                 + name + "</a>");
017         }
018     });
019 }
020 function getProducts(category_id){
021     $('.list-group-item').removeClass('active').addClass('');
022     $("#category_" + category_id).addClass('active');
023     $.getJSON("../ws/portal/products/" + category_id, function(result){
024         data = result.data;
025         $("#div_products").empty();
026         for(var row=0; row<data.length; row=row+1){
027             var id = data[row].id;
028             var name = data[row].name;
029             var published = data[row].published;
030             var icon = data[row].icon;
```

```
            var pricing = data[row].pricing;
            var short_description = data[row].short_description;
            var url = "../item.jsp?id=" + id;
            var item = '<div class="col-lg-4 col-md-6 mb-4">';
            item += '<div class="card h-100">';
            item += '<a id="link_title" href="' + url + '">';
            item += '<img class="card-img-top" src="../fotos/' + icon + '" alt="">';
            item += '</a>';
            item += '<div class="card-body">';
            item += '<h4 class="card-title">';
            item += '<a href="' + url + '">' + name + '</a>';
            item += '</h4>';
            item += '<h5>$' + pricing + '</h5>';
            item += '<p class="card-text">' + short_description + '</p>';
            item += '</div>';
            item += '<div class="card-footer">';
            item += '<small class="text-muted">';
            item += '&#9733; &#9733; &#9733; &#9733;&#9734;';
            item += '</small>';
            item += '</div>';
            item += '</div>';
            item += '</div>';
            $("#div_products").append(item);
        }
    });
}
```

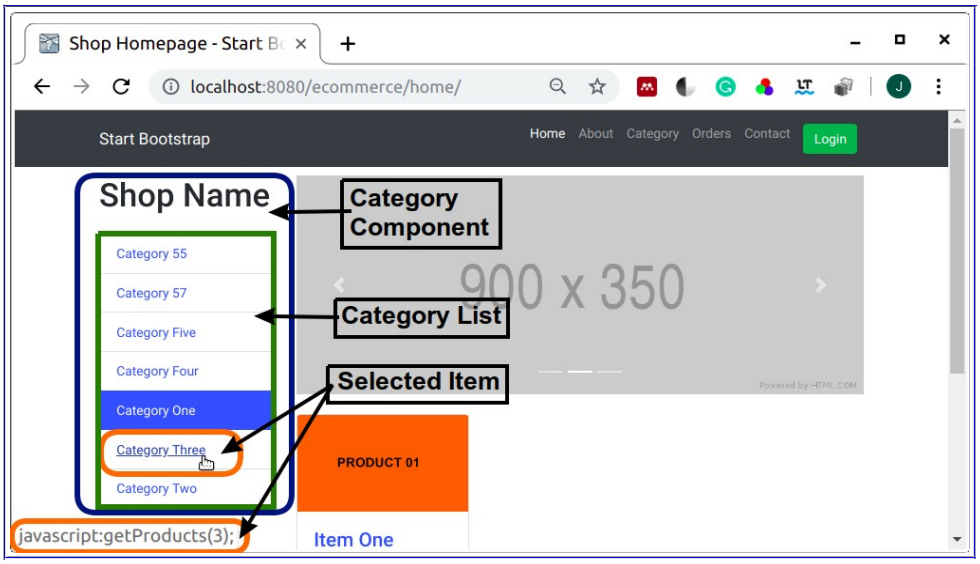
El archivo [home/index.jsp] incluye los nuevos componentes (category y products).

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
       pageEncoding="UTF-8"%>

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="viewport"
       content="width=device-width, initial-scale=1, shrink-to-fit=no">
<meta name="description" content="">
<meta name="author" content="">
<title>Shop Homepage - Start Bootstrap Template</title>
<!-- Bootstrap core CSS -->
<link rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
<!-- Custom styles for this template -->
<link href="../css/shop-homepage.css" rel="stylesheet">
</head>
<body>
    <!-- Navigation -->
    <jsp:include page="../portal/menu.jsp" />
    <!-- Page Content -->
    <div class="container">
        <div class="row">
            <div class="col-lg-3">
                <jsp:include page="../portal/category.jsp" />
            </div>
            <!-- /.col-lg-3 -->
            <div class="col-lg-9">
```

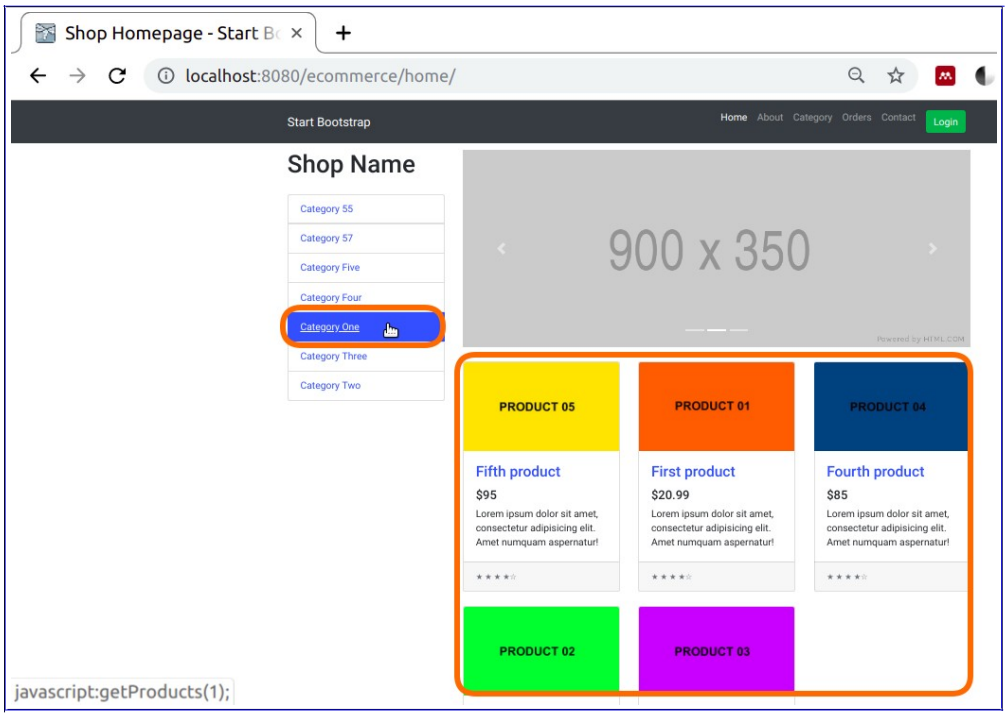
```
031                 <jsp:include page="../portal/carousel.jsp" />
032                 <jsp:include page="../portal/products.jsp" />
033             </div>
034             <!-- /.col-lg-9 -->
035         </div>
036         <!-- /.row -->
037     </div>
038     <!-- /.container -->
039     <!-- Footer -->
040     <jsp:include page="../portal/footer.jsp" />
041     <!-- Bootstrap core JavaScript -->
042     <script src="https://code.jquery.com/jquery-3.3.1.js"></script>
043     <script
044         src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js">
045     </script>
046     <script src="../js/portal.js"></script>
047     <script type="text/javascript">
048         getCategories(1);
049         getProducts(1);
050     </script>
051 </body>
052 </html>
```

6. PRUEBA DE LOS COMPONENTES WEB

Al llamar a la página home podemos ver el componente category, mostrando los registros de la tabla categories que tienen un valor de 1 en el campo published. Cuando se hace click en una categoría diferente a la actualmente seleccionada, el componente la resalta y muestra los registros de la tabla products que pertenecen a la categoría seleccionada.



Por defecto se muestran los productos de una categoría (1).



Al seleccionar una categoría se muestran los productos que le pertenecen en el componente products y se resalta la categoría seleccionada.