

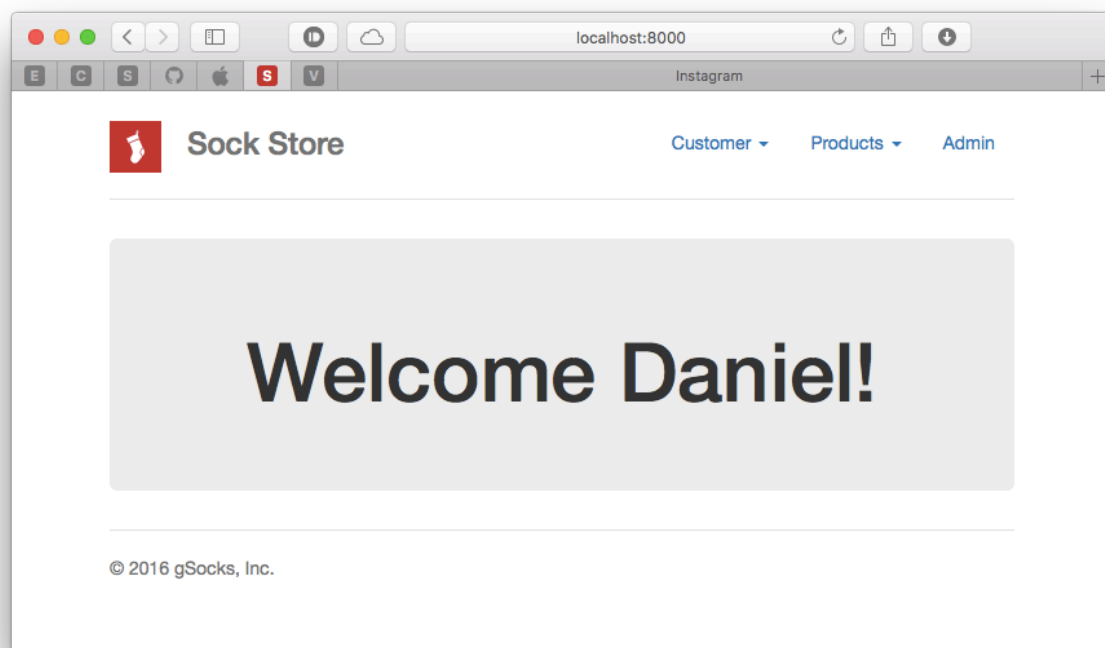
## Lab3.

---

- En el laboratorio anterior se creo el modelo del producto a vender en nuestro eCommerce.
- En esta entrega he:
  - Construido la vista de añadir productos y la vista de listado de productos.
  - Además de modificar el "index" de forma que dé la bienvenida al usuario.

## Bienvenida.

---



## Vista de añadir productos.

---

# Template.



The screenshot shows a code editor window titled "NewProduct.html" with a file explorer on the left. The file explorer shows a project structure with a "templates" folder containing "Base.html", "Home.html", "ListProducts.html", "NewCustomer.html", and "NewProduct.html". Below it is a "Store" folder containing various Python files. The main editor area shows the content of "NewProduct.html" with line numbers 1 through 12. The code is a Django template that extends "Base.html" and contains a form for adding a new product.

```
1 {% extends 'Base.html' %}
2
3 {% block content %}
4     <h1>{{title}}</h1><br>
5     <form method='POST' action=''>
6         {% csrf_token %}
7         {{ form.as_ul }}
8         <br><br>
9         <input type='submit' value='Send' class='btn btn-success
10             btn-block'>
11     </form>
12 {% endblock %}
```

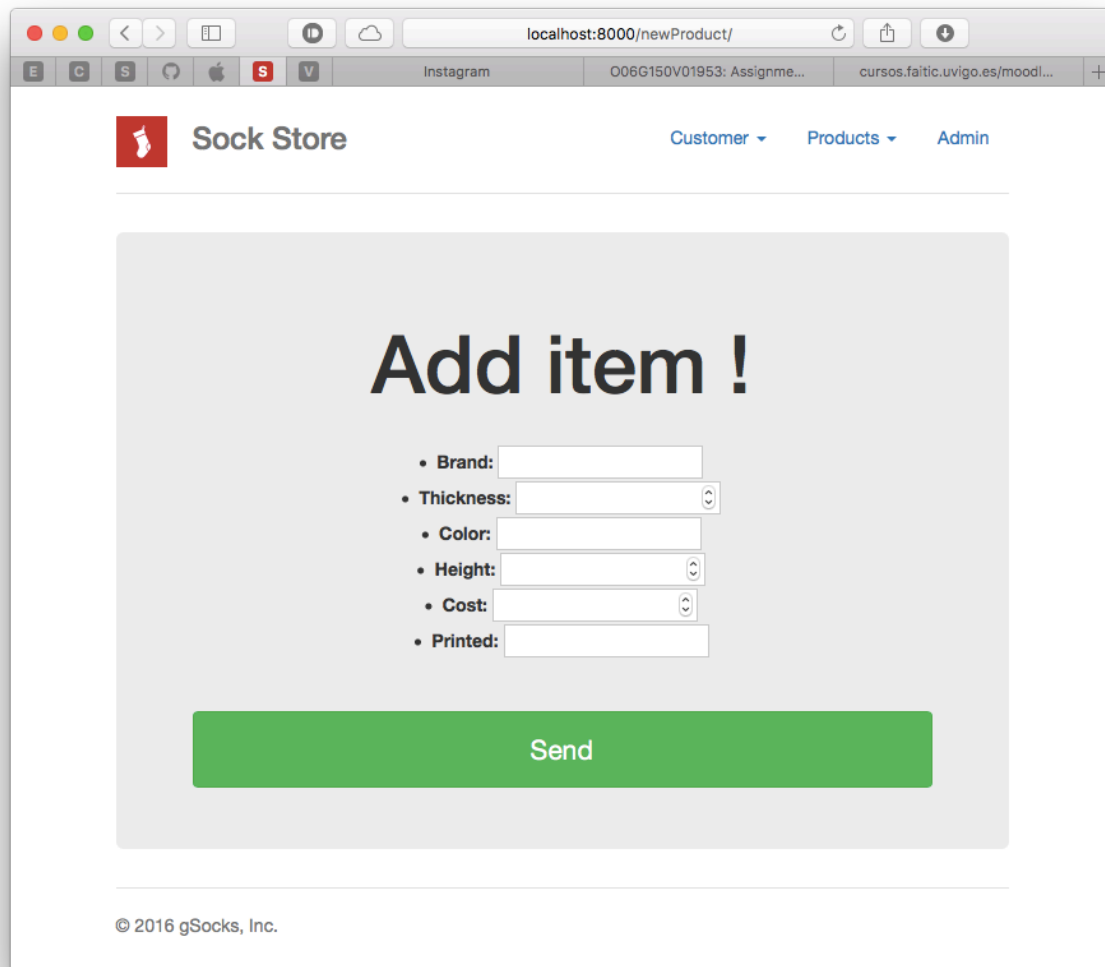
# View.



The screenshot shows a code editor window titled "views.py" with a file explorer on the left. The file explorer shows the same project structure as the previous screenshot, but with "views.py" selected in the "Store" folder. The main editor area shows the content of "views.py" with line numbers 33 through 49. The code defines a "newProduct" function that handles the POST request, saves the form, and renders the "NewProduct.html" template.

```
33 def newProduct(request):
34     form = ProductForm( request.POST or None )
35
36     if form.is_valid():
37         instance = form.save(commit = False)
38         print instance
39         instance.save()
40         return HttpResponseRedirect( '/newProduct' )
41
42     title = 'Add item !'
43     context={
44         'title':title,
45         'form':form,
46     }
47
48     return render(request,'NewProduct.html',context)
49
```

## Render.



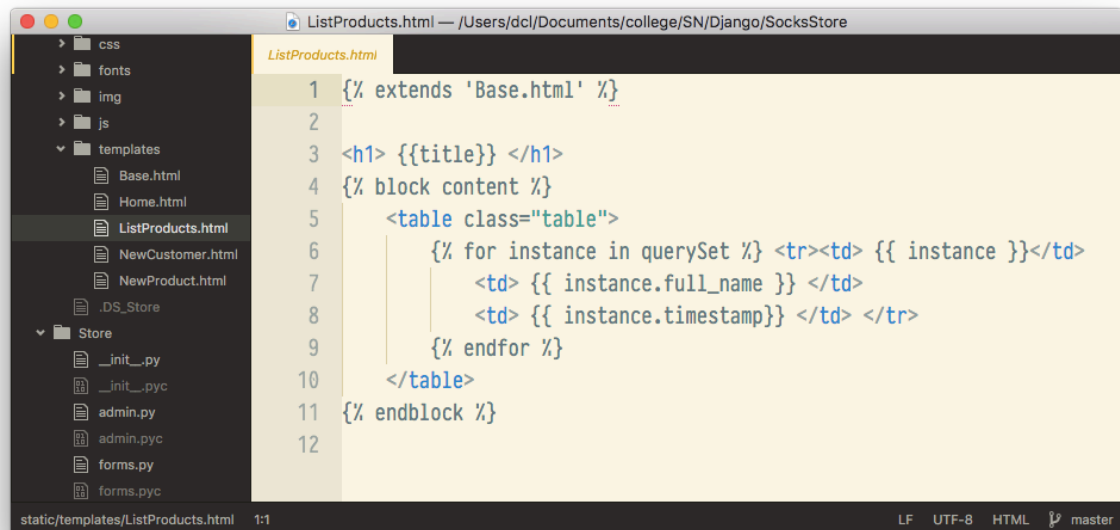
The screenshot shows a web browser window with the address bar at `localhost:8000/newProduct/`. The browser tabs include 'Instagram', 'O06G150V01953: Assignme...', and 'cursos.faltec.uvigo.es/moodl...'. The page header for 'Sock Store' features a red sock icon, the text 'Sock Store', and navigation links for 'Customer', 'Products', and 'Admin'. The main content area is titled 'Add item !' and contains a form with the following fields:

- Brand:
- Thickness:
- Color:
- Height:
- Cost:
- Printed:

A large green 'Send' button is positioned below the form fields. At the bottom of the page, the copyright notice '© 2016 gSocks, Inc.' is displayed.

## Vista de listar productos.

# Template.

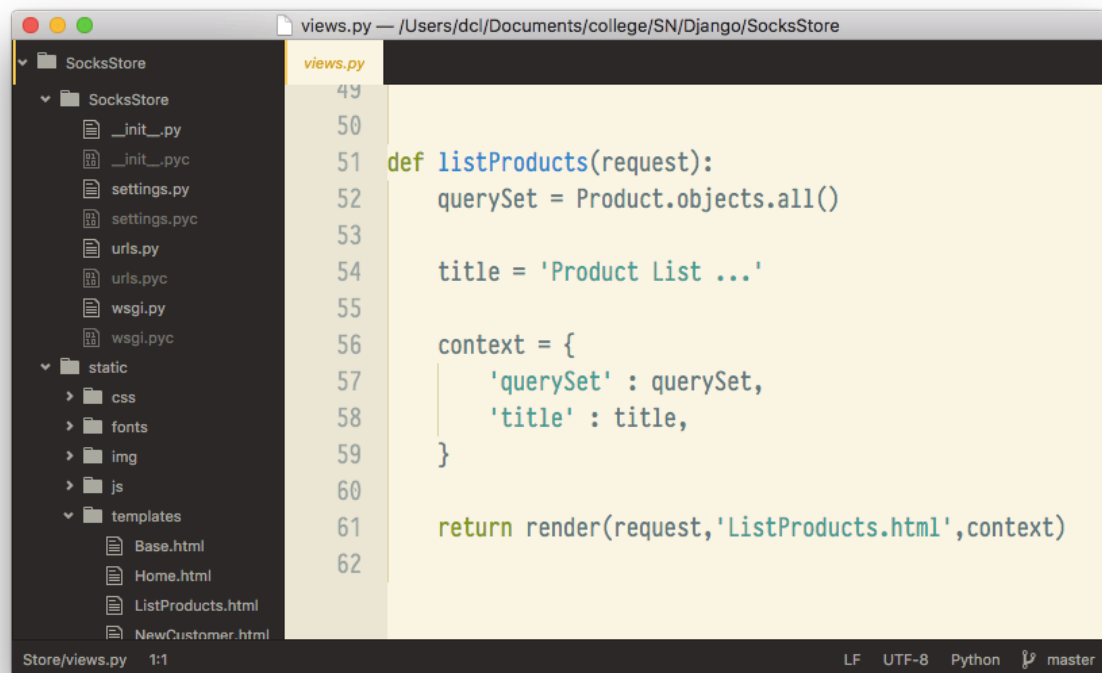


The screenshot shows a code editor window titled "ListProducts.html" with a file explorer on the left. The file explorer shows a project structure with folders like "css", "fonts", "img", "js", "templates", and "Store". The "templates" folder is expanded, showing files like "Base.html", "Home.html", "ListProducts.html", "NewCustomer.html", and "NewProduct.html". The "ListProducts.html" file is selected and its content is displayed in the editor. The code is a Django template that extends "Base.html" and contains a table to list products. The table has columns for the product instance, its full name, and its timestamp. The code is as follows:

```
1 {% extends 'Base.html' %}
2
3 <h1> {{title}} </h1>
4 {% block content %}
5     <table class="table">
6         {% for instance in querySet %} <tr><td> {{ instance }}</td>
7         <td> {{ instance.full_name }} </td>
8         <td> {{ instance.timestamp}} </td> </tr>
9         {% endfor %}
10    </table>
11 {% endblock %}
12
```

The status bar at the bottom indicates the file is located at "static/templates/ListProducts.html" and is 1:1 lines long. The encoding is UTF-8 and the language is HTML.

# View.



The screenshot shows a code editor window titled "views.py" with a file explorer on the left. The file explorer shows a project structure with folders like "SocksStore", "static", "css", "fonts", "img", "js", and "templates". The "SocksStore" folder is expanded, showing files like "\_\_init\_\_.py", "settings.py", "urls.py", "wsgi.py", and "views.py". The "views.py" file is selected and its content is displayed in the editor. The code is a Django view function named "listProducts" that takes a request object as an argument. It queries the database for all product objects and renders the "ListProducts.html" template with the querySet and a title. The code is as follows:

```
49
50
51 def listProducts(request):
52     querySet = Product.objects.all()
53
54     title = 'Product List ...'
55
56     context = {
57         'querySet' : querySet,
58         'title' : title,
59     }
60
61     return render(request, 'ListProducts.html', context)
62
```

The status bar at the bottom indicates the file is located at "Store/views.py" and is 1:1 lines long. The encoding is UTF-8 and the language is Python.

# Render.

