**Instalando el entorno virtual**

C:\Users\Armando>pip3 install virtualenv

Ingresando a D

C:\>d:

**Creando la Carpeta misproyectos en la raiz D**

D:\>mkdir misproyectos

**Ingresando a Ella**

D:\>cd misproyectos

**Creando la SubCarpeta ProjectoBarberia**

D:\misproyectos>mkdir ProjectoBarberia

**Ingresando a Ella**

D:\misproyectos>cd ProjectoBarberia

D:\misproyectos\ProjectoBarberia>

**Instalando el entorno virtual en mi proyecto barberia**

D:\misproyectos\ProjectoBarberia>virtualenv venv

Using base prefix 'c:\\users\\armando\\appdata\\local\\programs\\python\\python37-32'

New python executable in D:\misproyectos\ProjectoBarberia\venv\Scripts\python.exe

Installing setuptools, pip, wheel...

done.

**Activando el entorno virtual**

D:\misproyectos\ProjectoBarberia>venv\Scripts\activate

(venv) D:\misproyectos\ProjectoBarberia>

**DesActivando el entorno virtual**

(venv) D:\misproyectos\ProjectoBarberia\venv\Scripts>deactivate.bat

D:\misproyectos\ProjectoBarberia\venv\Scripts>

**Instalado Django**

(venv) D:\misproyectos\ProjectoBarberia>pip3 install django

Collecting django

Using cached https://files.pythonhosted.org/packages/54/85/0bef63668fb170888c1a2970ec897d4528d6072f32dee27653381a332642/Django-2.2-py3-none-any.whl

**Iniciando el Proyecto de Barberia**

(venv) D:\misproyectos\ProjectoBarberia>django-admin startproject ProjectoBarberia

(venv) D:\misproyectos\ProjectoBarberia>

**Ejecutando el Proyecto de Barberia**

**Ingresamos al proyecto**

(venv) D:\misproyectos\ProjectoBarberia>cd ProjectoBarberia

**Y lo Ejecutamos**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>python manage.py runserver

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

You have 17 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.

Run 'python manage.py migrate' to apply them.

April 07, 2019 - 18:10:28

Django version 2.2, using settings 'ProjectoBarberia.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL-BREAK.

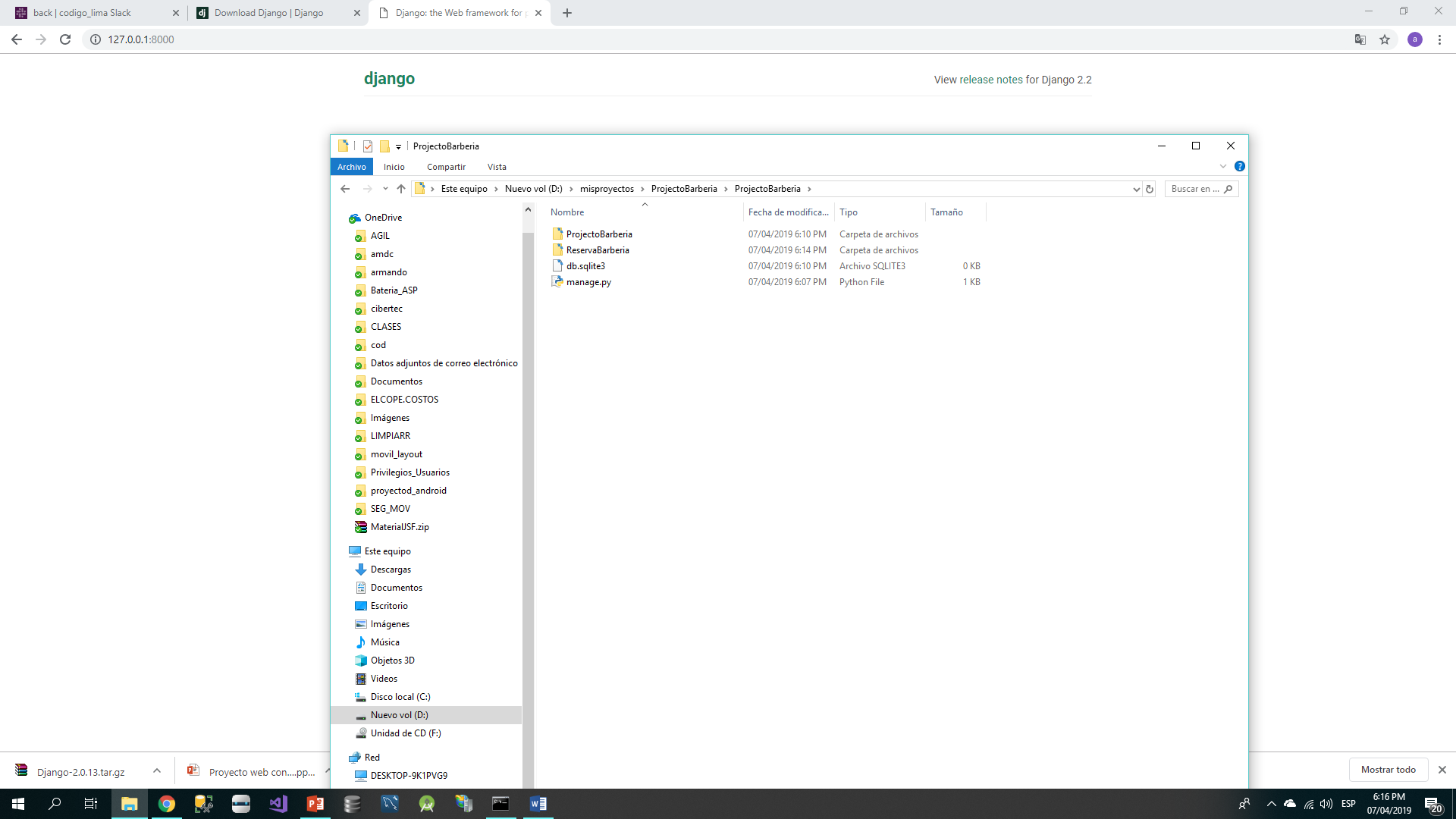
**Presione Ctrl + Break (Ctrl +Fin+Pausa (esta en la tecla REPAG) Para detener el servicio de desarrollo**

**Creando Nuestro primer Hola Mundo**

**Creando nuestra aplicación de ReservaBarberia dentro del projecto principal para eso utilizo startapp**

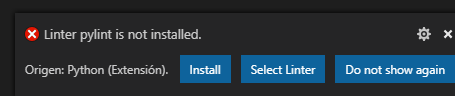
(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>django-admin startapp ReservaBarberia

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>

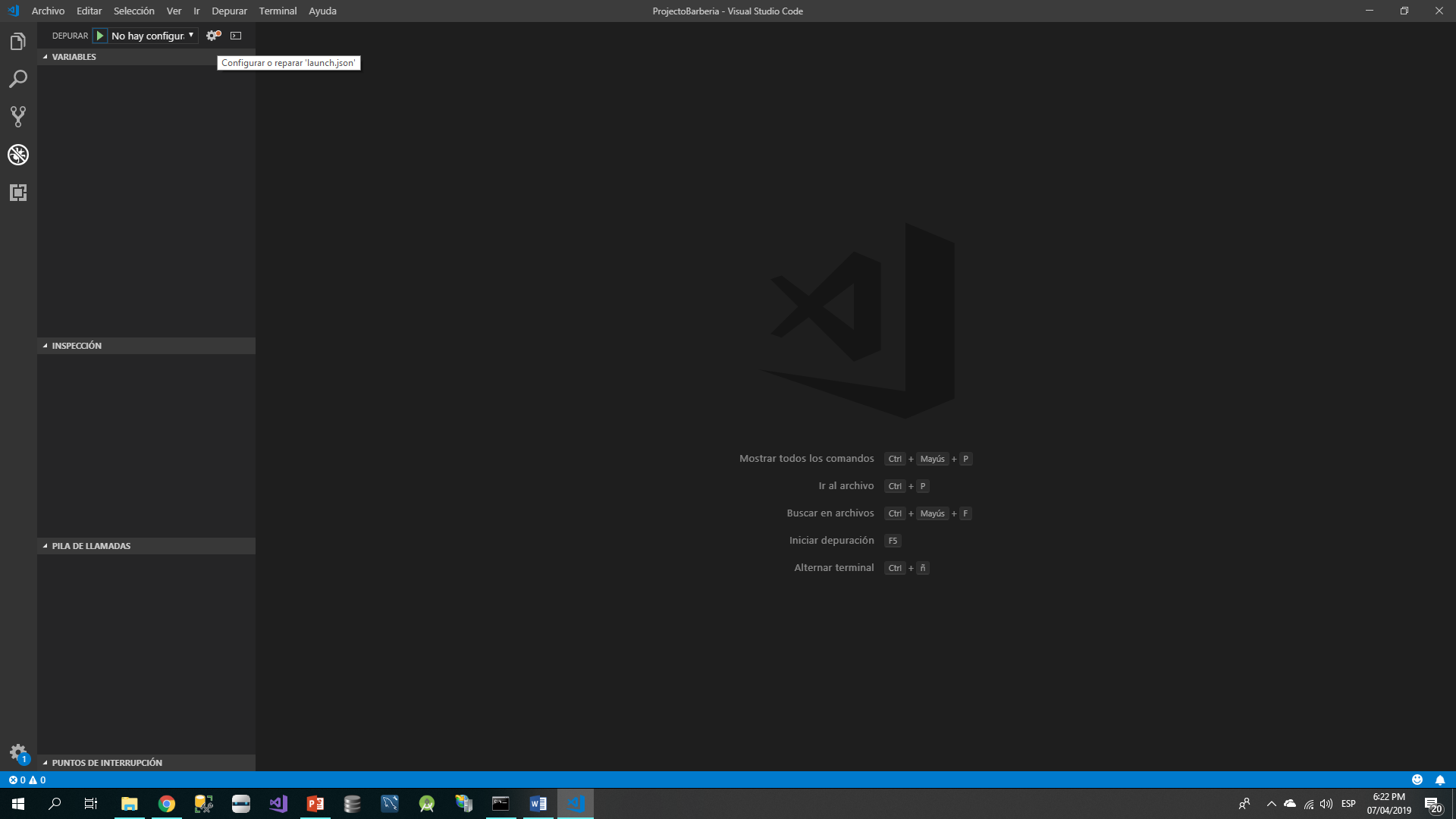


**Abrir el entorno virtual en Visual Studio Code**

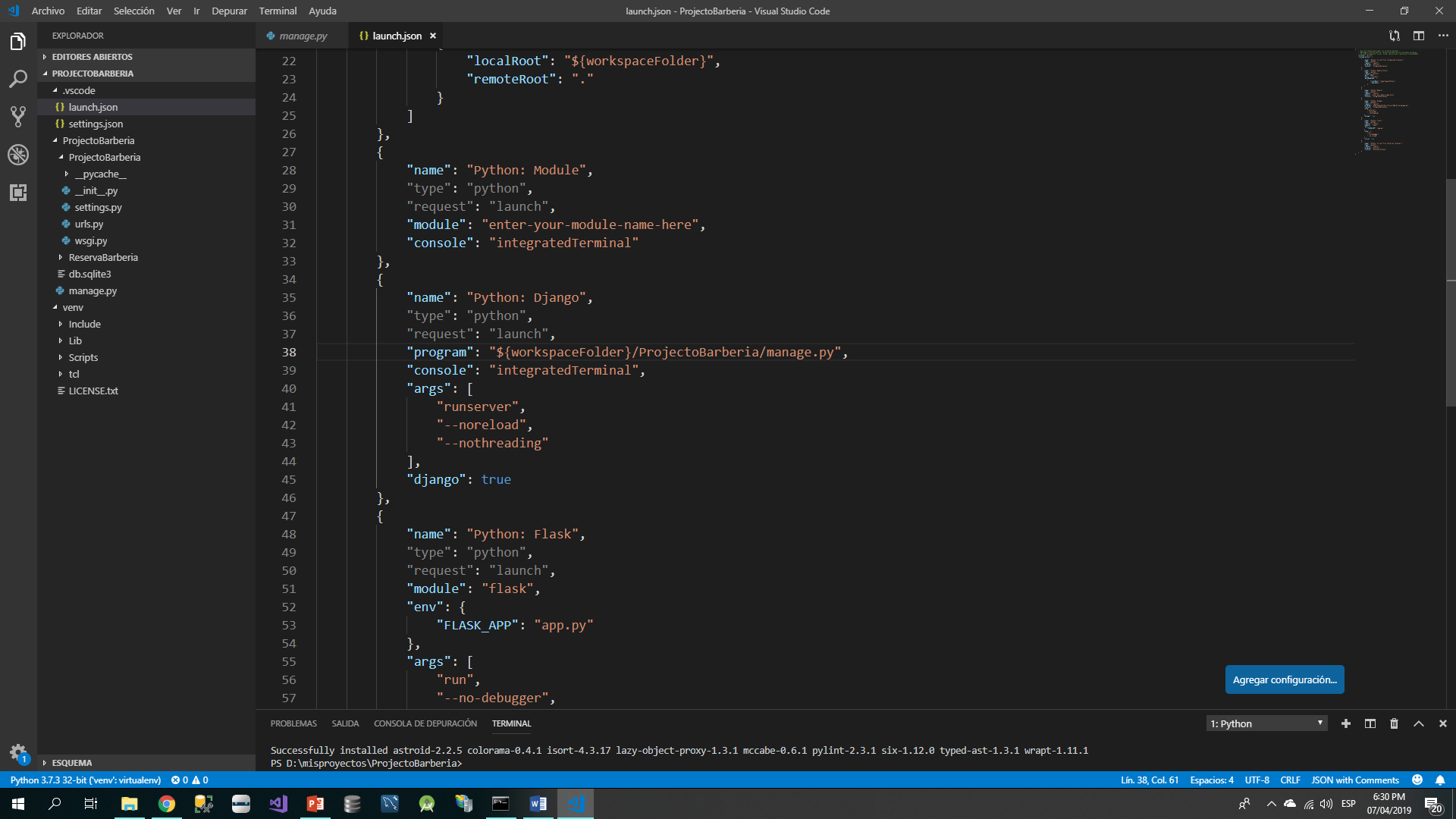
1. Arrastre la carpeta raiz hacia el ID
2. Ir al archivo manage.py del proyecto “ProjectoBarberia” e instale :



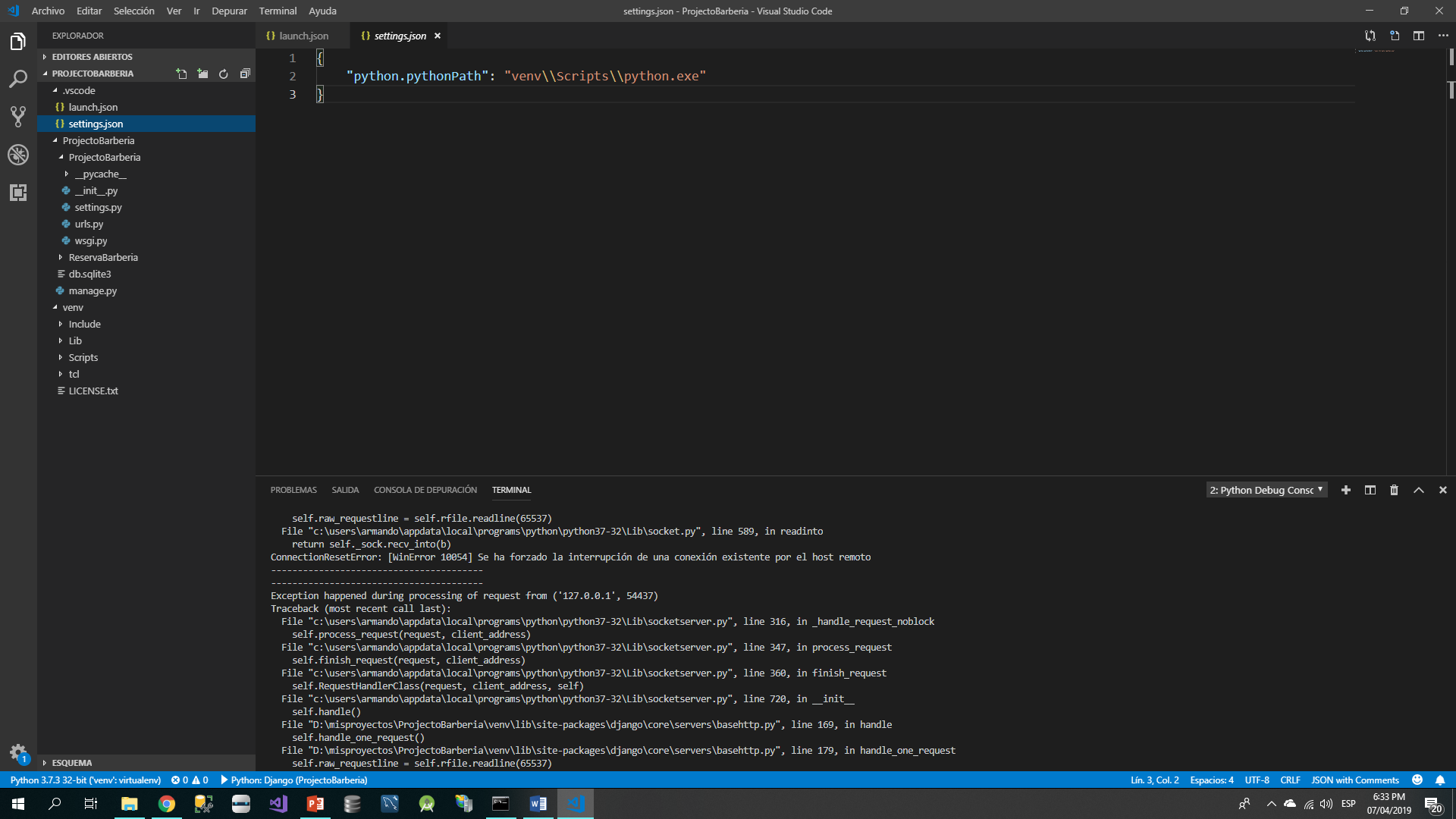
1. Presione clik en la herramienta depurar
2. Presione clik en configurar o reparar lauch.json



1. Busque la configuracion de “Python:Django”
2. Agregue dentro del path el nombre de su carpeta de proyecto principal



1. El archivo settings.json contiene la ruta de su proyecto virtual a ejecutarse



1. Abrimos el archivo settings del proyecto raiz “ProjectoBarberia” y le agregamos la nueva aplicación que hemos creado:
2. INSTALLED\_APPS = [
3. 'django.contrib.admin',
4. 'django.contrib.auth',
5. 'django.contrib.contenttypes',
6. 'django.contrib.sessions',
7. 'django.contrib.messages',
8. 'django.contrib.staticfiles',
9. 'ReservaBarberia'
10. ]
11. Abrimos el archivo views.py de nuestro proyecto “ReservaBarberia” y agregamos el siguiente codigo:
12. from django.shortcuts import render
13. from django.http import HttpResponse
14. def home (request):
15. return HttpResponse('Django vive en codigo')
16. # Create your views here.
17. Ahora modificamos el archivo urls.py del proyecto principal “ProyectoBarberia” y agregamos nuestra nueva vista:
18. from django.contrib import admin
19. from django.urls import path
20. from ReservaBarberia import views
21. urlpatterns = [
22. path('admin/', admin.site.urls),
23. path('',views.home,name='home')
25. ]

**Uso de Plantillas**

1. Creamos la carpeta templates dentro del proyecto principal “ProyectoBarberia”
2. Dentro de ficha carpeta creamos un archivo llamado base.html
3. Digitamos en siguiente html en su contenido:
4. <!DOCTYPE html>
5. <html lang="en">
6. <head>
7. <meta charset="UTF-8">
8. <title>Home</title>
9. </head>
10. <h1>Hola Django</h1>
11. </html>
12. En el archivo settings colocar el path de la carpeta donde se guardara las plantillas
13. TEMPLATES = [
14. {
15. 'BACKEND': 'django.template.backends.django.DjangoTemplates',
16. 'DIRS': [os.path.join(BASE\_DIR, 'templates')],
17. 'APP\_DIRS': True,
18. Abrimos el archivo views.py de nuestro proyecto “ReservaBarberia” y agregamos el siguiente codigo:
19. from django.shortcuts import render
20. from django.template import RequestContext
21. def home(request):
22. context = {'foo': 'bar'}
23. return render(request,'base.html', context)

**Pasando parametros en plantillas**

1. Abrimos el archivo views.py de nuestro proyecto “ReservaBarberia” y aumentamos el siguiente codigo:
2. def ejemplo(request):
3. context = {'foo': 'bar'}
4. titulo = 'Codigo Vive Django'
5. nombre = 'Pasando Variables en Django'
6. lista = [2,3,5,65,87,4,6,321,654,9]
7. return render(request,'ejemplo.html',{'title':titulo, 'nombre':nombre ,'lista' :lista},context)
8. Dentro de la carpeta template creamos el archivo ejemplo.html
9. <!DOCTYPE html>
10. <html lang="en">
11. <head>
12. <meta charset="UTF-8">
13. <title>{{ title }}</title>
14. </head>
15. <body>
16. <h1>{{ nombre }}</h1>
17. <ul>
18. {% for l in lista %}
19. <li>{{ l }}</li>
20. {% endfor %}
21. </ul>
22. </body>
23. </html>
24. Modificamos el urls.py de nuestro proyecto principal para que se muestre en otra direccion el nuevo html
25. urlpatterns = [
26. path('admin/', admin.site.urls),
27. path('',views.home,name='home'),
28. path('ejemplo/variables',views.ejemplo,name='ejemplo'),
30. ]
31. Digite la siguiente direccion desde su browser : <http://127.0.0.1:8000/ejemplo/variables>

**Crear el Superusuario de Administrador e Ingresar a el**

**Debera migrar todas las tablas que creo en el modelo**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py migrate**

Operations to perform:

**Crear el SuperUsuario**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py createsuperuser**

Username (leave blank to use 'armando'): aruiz

Email address: armando.eu.ruiz@gmail.com

Password:

Password (again):

Superuser created successfully.

**Inicie su proyecto**

**Ingresar a la siguiente Direccion :** [http://127.0.0.1:8000/admin](http://127.0.0.1:8000/admin/login/?next=/admin/)

Ingrese sus credenciales que creo en el superusuario

**Modificar Su Clave de super usuario**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py changepassword** aruiz

Changing password for user 'aruiz'

Password:

Password (again):

Password changed successfully for user 'aruiz'

**Cambiar de Clave desde la consola**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py shell**

Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32

Type "help", "copyright", "credits" or "license" for more information.

(InteractiveConsole)

>>> **from django.contrib.auth.models import User**

>>> **u = User.objects.get(username='aruiz')**

>>> **u.set\_password('aruiz2019')**

>>> **u.save()**

>>>**quit()**

**Ver tos los Usuarios Actuales**

>>> **from django.contrib.auth.models import User**

>>> **usuarios = User.objects.all()**

>>> **usuarios**

<QuerySet [<User: aruiz>]>

**Crear Nuestro Primer Modelo**

1. SI salio de su proyecto vuelva a activarlo ingresando los siguientes pasos:

D:\>**cd misproyectos**

D:\misproyectos>**cd ProjectoBarberia**

D:\misproyectos\ProjectoBarberia>**venv\Scripts\activate**

(venv) D:\misproyectos\ProjectoBarberia>**cd ProjectoBarberia**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py runserver**

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

April 09, 2019 - 09:47:12

Django version 2.2, using settings 'ProjectoBarberia.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL-BREAK.

Ingresar al archivo models.py de la aplicación “ReservaBarberia” y crear el siguiente modelo:

1. from django.db import models
2. class reserva(models.Model):
3. id =models.AutoField(primary\_key = True)
4. barbero = models.CharField(max\_length = 150 ,blank = False , null = False)
5. cliente = models.CharField(max\_length = 150 ,blank = False , null = False)
6. servicio = models.CharField(max\_length = 150 ,blank = False , null = False)
7. fecha = models.DateField()
8. obs = models.TextField
10. # Create your models here.
11. Agregar el siguiente comando para migrar el modelo:

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py makemigrations**

Migrations for 'ReservaBarberia':

ReservaBarberia\migrations\0001\_initial.py

- Create model reserva

1. Pasaremos ahora a migrarlo a la Base de datos:

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py migrate**

Operations to perform:

Apply all migrations: ReservaBarberia, admin, auth, contenttypes, sessions

Running migrations:

Applying ReservaBarberia.0001\_initial... OK

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>

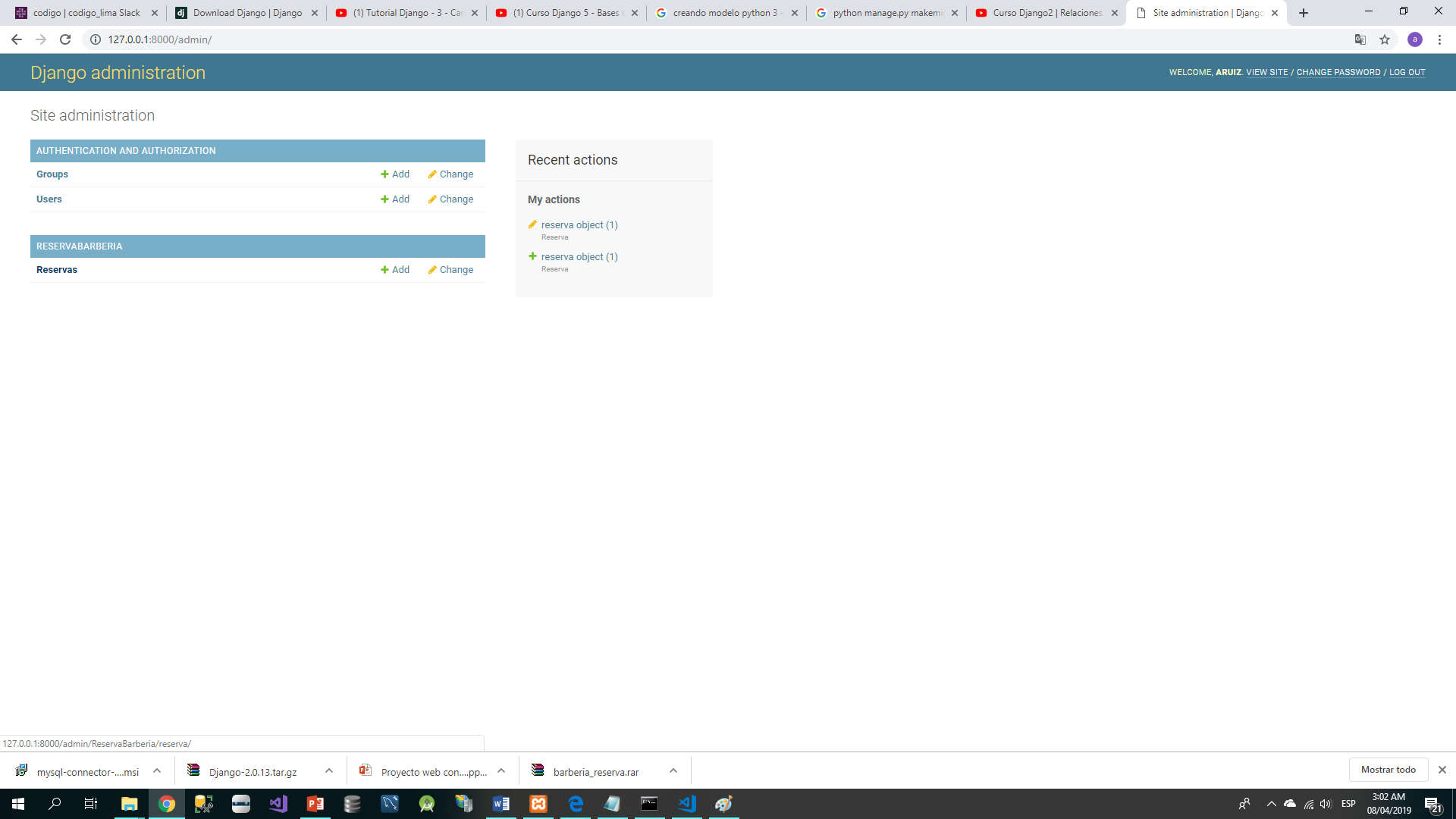
1. Dirigirse al archivo admin.py de la carpeta migrations del app “ReservaBarberia” y agregar lo siguiente:
2. from django.contrib import admin
3. from .models import reserva
4. admin.site.register(reserva)
5. # Register your models here.
6. Ejecute nuevamente el Servidor:

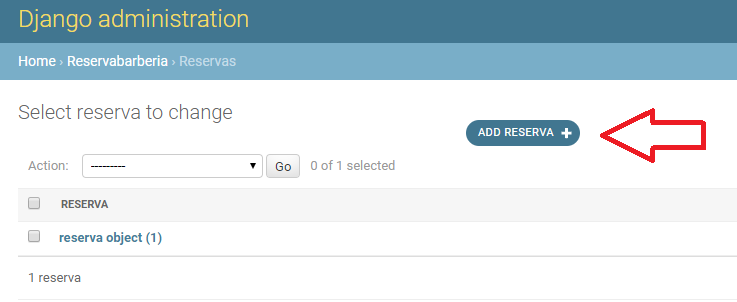
(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py runserver**

Watching for file changes with StatReloader

Performing system checks...

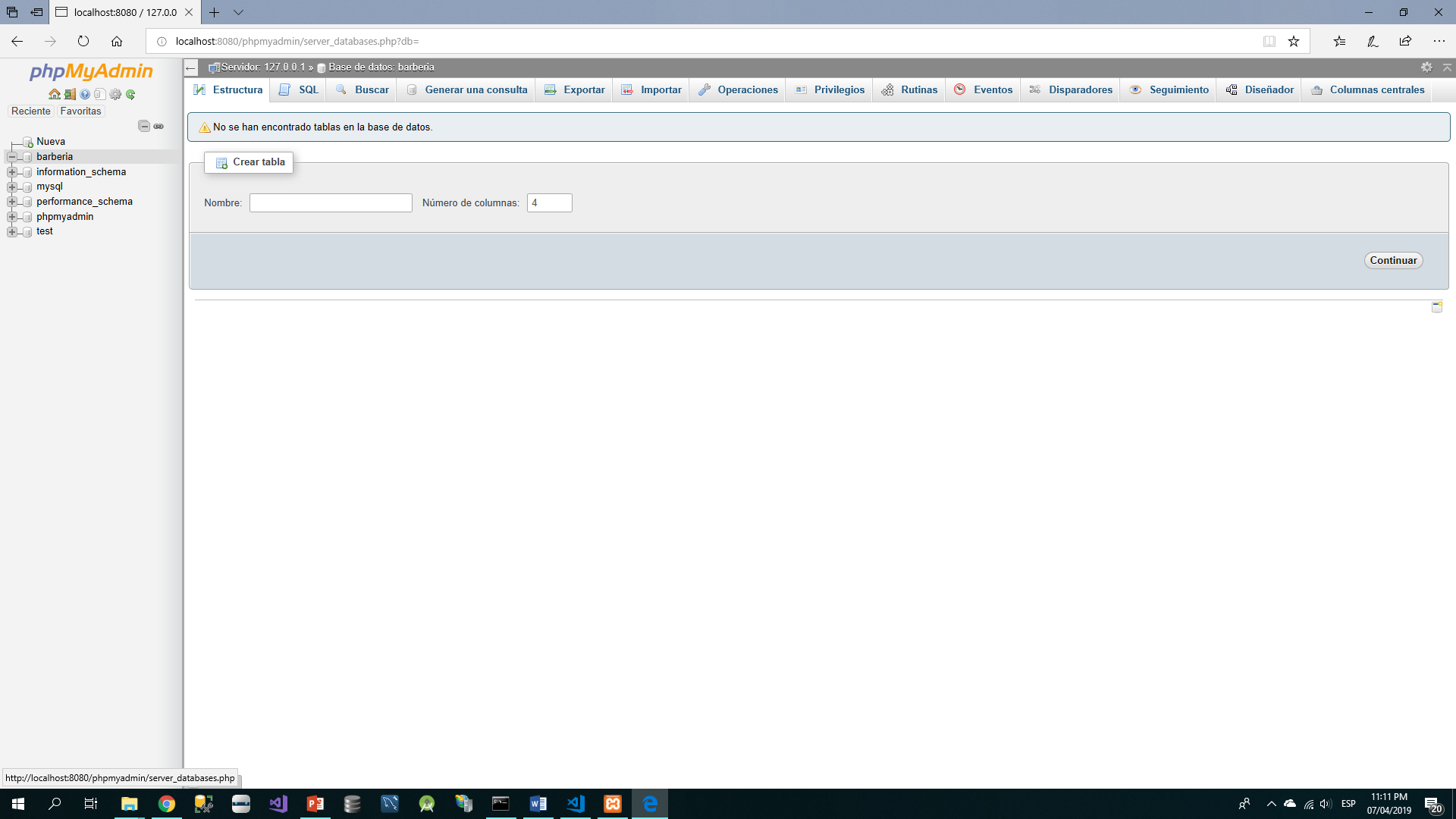
1. Ingrese a la pagina de administrador y podra hacer un crud de su tabla:



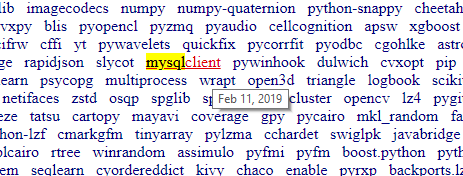


**Conectarlo con Mysql**

1. Ingresar a XAMPP y crear la bd barberia:



1. Ingresar a la pagina : <https://www.lfd.uci.edu/~gohlke/pythonlibs/> para descargarse las librerias
2. Presional Ctrl + F en la pagina mostrada y escribir mysql
3. Pulsar clik en la libreria :



1. Descarguese el archivo whl de acuerdo a la version de python instalado para eso en su proyecto activo digite:

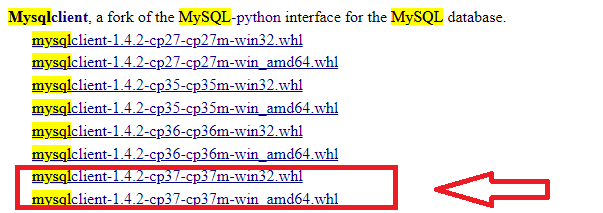
(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python**

**Python 3.7.3** (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32

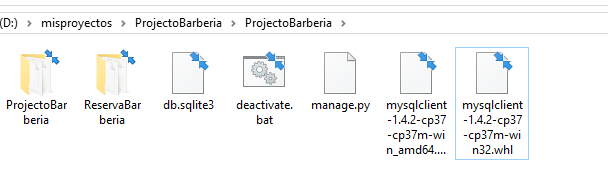
Type "help", "copyright", "credits" or "license" for more information.

>>>

1. Como vera la version de mi pythones Python 3.7.3 descargarse los siguientes librerias la de 32 y 64 bits:



1. Copiar dichos archivos en la carpeta donde se encuentra el archivo manage.py de su proyecto



1. Salir del shell de python con quit() y esto le llevara nuevamente a su proyecto virtual:

Type "help", "copyright", "credits" or "license" for more information.

**>>> quit()**

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>

1. Instale la librería la de 64 y 32 bits:

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**pip install mysqlclient-1.4.2-cp37-cp37m-win32.whl**

1. Ingrese al archivo settings de su proyecto principal y modifique la cadena de conexión:
2. DATABASES = {
3. 'default': {
4. 'ENGINE': 'django.db.backends.mysql',
5. 'NAME': 'barberia',
6. 'USER': 'root',
7. 'PASSWORD': '',
8. 'HOST': 'localhost',
9. 'PORT': '3306',
10. 'OPTIONS': {
11. 'init\_command': "SET sql\_mode='STRICT\_TRANS\_TABLES'",
12. },
13. }
14. }
15. Migre todos los cambios a MYSQL con el siguiente comando:

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py migrate**

**System check identified some issues:**

WARNINGS:

?: (mysql.W002) MySQL Strict Mode is not set for database connection 'default'

HINT: MySQL's Strict Mode fixes many data integrity problems in MySQL, such as data truncation upon insertion, by escalating warnings into errors. It is strongly recommended you activate it. See: https://docs.djangoproject.com/en/2.2/ref/databases/#mysql-sql-mode

Operations to perform:

Apply all migrations: ReservaBarberia, admin, auth, contenttypes, sessions

Running migrations:

No migrations to apply.

**Modelos en SQLITE**

**Cadena de Conexión:**

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),

}

}

from django.db import models

class Barbero(models.Model):

codbarbero =models.AutoField(primary\_key = True)

barberonom = models.CharField('Barbero',max\_length = 150 ,blank = False , null = False)

fecharegistro = models.DateField()

descripcion = models.TextField(null = True)

class Meta:

verbose\_name = 'Barbero'

verbose\_name\_plural= 'Barberos'

ordering= ['barberonom']

def \_\_str\_\_(self):

return self.barberonom

class Cliente(models.Model):

codcliente =models.AutoField(primary\_key = True)

clientenom = models.CharField('Cliente',max\_length = 150 ,blank = False , null = False)

fecharegistro = models.DateField()

email =models.EmailField('Correo',max\_length = 254)

telefono =models.CharField('Telefono',max\_length = 254)

class Meta:

verbose\_name = 'Cliente'

verbose\_name\_plural= 'Clientes'

ordering= ['clientenom']

def \_\_str\_\_(self):

return self.clientenom

class Servicio(models.Model):

codservicio =models.AutoField(primary\_key = True)

servicionom = models.CharField('Servicio',max\_length = 150 ,blank = False , null = False)

nomcorto = models.CharField('Abrev',max\_length = 50 ,blank = False , null = False)

fecharegistro = models.DateField()

descripcion = models.TextField(null = True)

class Meta:

verbose\_name = 'Servicio'

verbose\_name\_plural= 'Servicios'

ordering= ['servicionom']

def \_\_str\_\_(self):

return self.servicionom

class Producto(models.Model):

codproducto =models.AutoField(primary\_key = True)

productonom = models.CharField('Producto',max\_length = 150 ,blank = False , null = False)

nomcorto = models.CharField('Abrev',max\_length = 50 ,blank = False , null = False)

fecharegistro = models.DateField()

descripcion = models.TextField(null = True)

class Meta:

verbose\_name = ‘Producto’

verbose\_name\_plural= ‘Productos’

ordering= [‘productonom’]

def \_\_str\_\_(self):

return self.productonom

class Reserva(models.Model):

codreserva = models.AutoField(primary\_key = True)

codbarbero = models.ForeignKey(Barbero,on\_delete = models.CASCADE)

codcliente = models.ForeignKey(Cliente,on\_delete = models.CASCADE)

codservicio = models.ManyToManyField(Servicio)

fechareserva = models.DateField()

obs = models.TextField(null = True)

**Para la relacion de Uno a muchos se necesita una clave foranea**

barbero = models.ForeignKey(Barbero,on\_delete = models.CASCADE)

**de Muchos a Muchos**

barbero = models.ManyToManyField(Barbero,on\_delete = models.CASCADE)

admin.py

from django.contrib import admin

from .models import Reserva,Barbero,Cliente,Servicio,Producto

admin.site.register(Barbero)

admin.site.register(Cliente)

admin.site.register(Servicio)

admin.site.register(Producto)

admin.site.register(Reserva)

Ahora debera migrar los modelos a la bd:

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py makemigrations ReservaBarberia**

Migrations for ‘ReservaBarberia’:

ReservaBarberia\migrations\0001\_initial.py

- Create model barberos

- Create model clientes

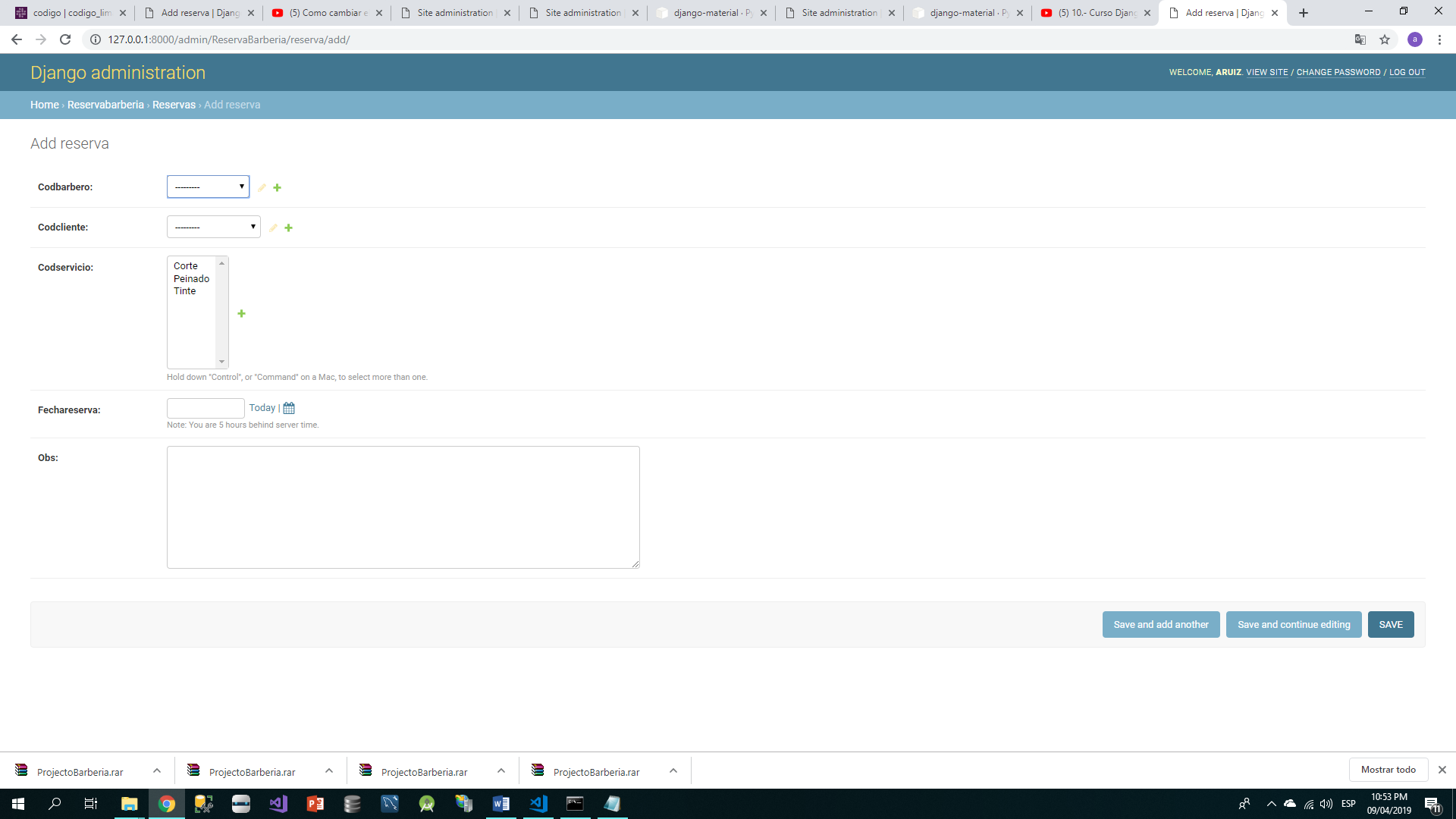
- Create model productos

- Create model reserva

- Create model servicios

(venv) D:\misproyectos\ProjectoBarberia\ProjectoBarberia>**python manage.py migrate**

1. Si es necesario deberar crer un SuperUsuario si la tabla esta vacia para ingresar al administrador:
2. Este es la apariencia de nuestra vista en el administrador



**Uso de Herencia de Plantillas (Master Page)**

1. En la raiz principal crear la carpeta template donde se guardara las plantillas de su proyecto
2. Dentro de dicha carpeta crear la subcarpeta Base
3. Agregar en el archivo settings.py la siguiente direccion donde django tomara las plantillas
4. 'DIRS': [os.path.join(BASE\_DIR, 'templates')],
5. Dentro de la subcarpeta Base crear el archivo base.html
6. Dentro del archivo escribir la palabra html:5 y pulsar enter para crear la estructura del documento

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>{% block title %} Document {% endblock %}</title>

</head>

<body>

{% block header %}

<h4>Este es el bloque del header base</h4>

{% endblock %}

{% block content %}

<h5>Este es el contenido del archivo base</h5>

{% endblock %}

{% block footer %}

<h5>Este es el footer del base</h5>

{% endblock %}

</body>

</html>

1. Dentro de la carpeta templates crear la subcarpeta llamada reservas en ella iran todos los templates para el app de reservas
2. Dentro de dicha subcarpeta crear el archivo llamado index.html y digite lo siguiente para heredar el contenido de la plantilla base

{% extends 'base/base.html' %}

1. En el archivo view.py de su app ReservaBarberia agregue la siguiente definicion:
2. def index(request):
3. return render(request,'reservas/index.html')
4. En el archivo url.py del proyecto principal agrege la siguiente path marcado de rojo :

from django.contrib import admin

from django.urls import path

from ReservaBarberia import views

urlpatterns = [

path('admin/', admin.site.urls),

path('',views.home,name='home'),

path('inicio/',views.index,name='index')

]

Ejecute el servidor y digite lo siguiente: <http://127.0.0.1:8000/inicio/>

1. Ahora si queremos sobreesbir nuestra plantilla base deberemos digitar lo siguiente en el archivo index.html de la subcarpeta reservas:

{% extends 'base/base.html' %}

{% block header %}

<h5>este es un nuevo header del index de nuestra app de reservas</h5>

{% endblock %}

{% block content %}

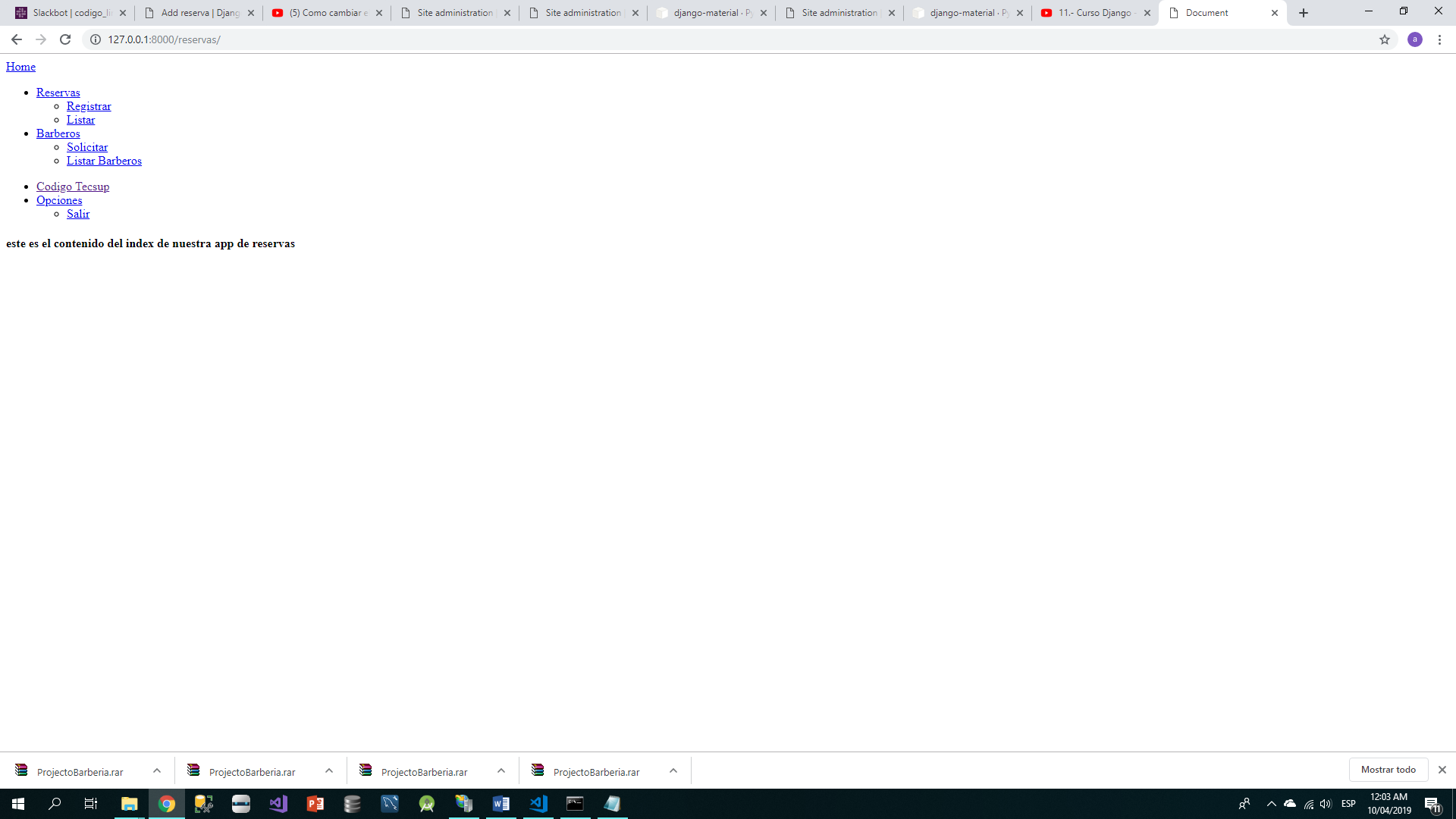
<h5>este es el contenido del index de nuestra app de reservas</h5>

{% endblock %}

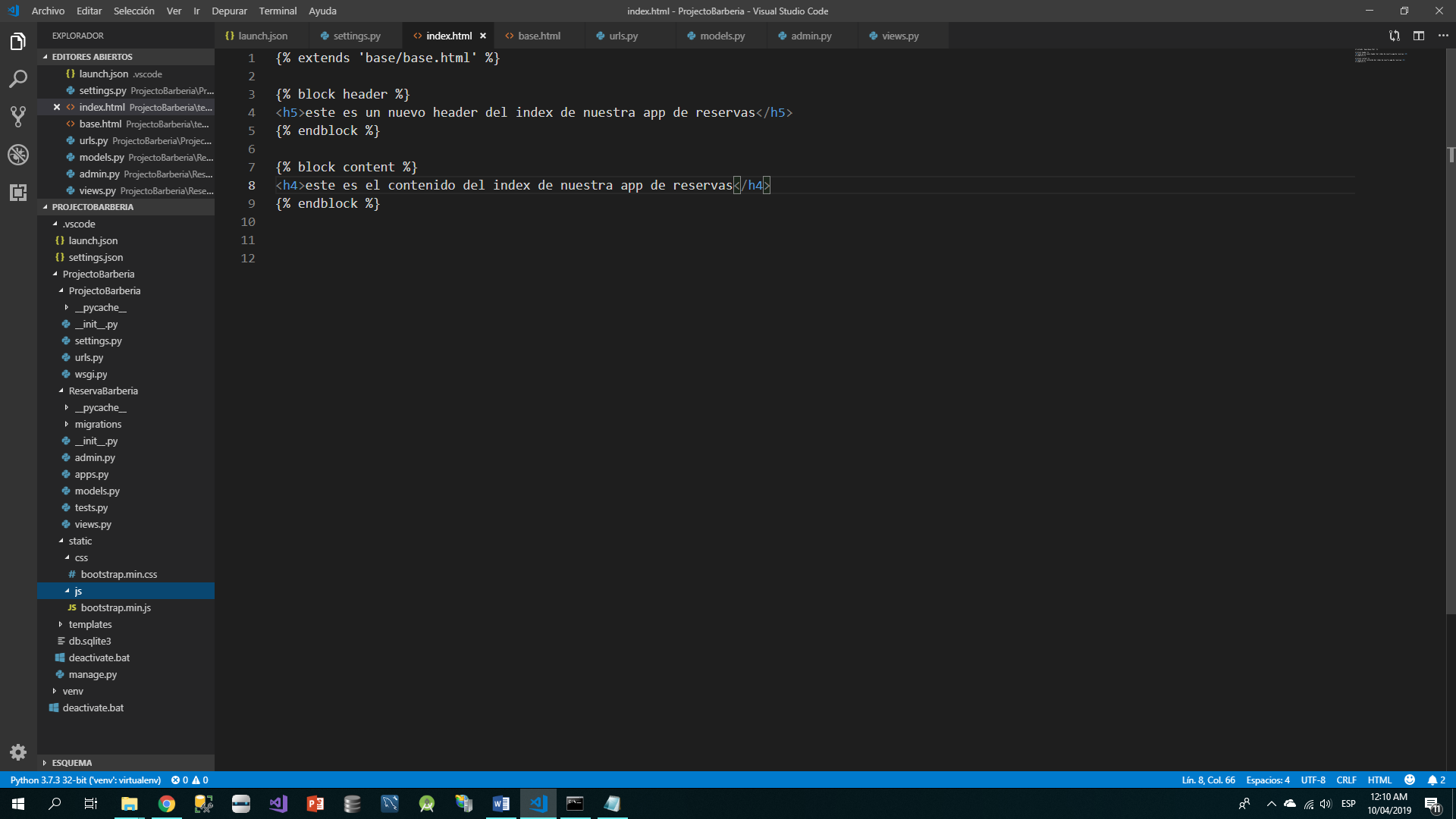
1. Actualize su pagina y veras que hemos sobreescrito el bloque del header

**Archivos estaticos**

1. Actualizar el archivo base.html dado en la practica
2. Llevantar el servidor y digite la siguiente direccion : <http://127.0.0.1:8000/inicio/> debera tener la siguiente apariencia:



1. En la carpeta principal ProjectoBarberia crear una carpeta llamada static el cual tendra nuestros css,js,imágenes etc , debera estar al mismo nivel que template
2. Dentro de dicha carpeta crear las subcarpetas: CSS, JS
3. Copiar los archivos de bootstratp dados en la practica a sus respectivas carpetas



1. Abrir el archivo setting.py del proyecto principal y agregar la siguiente linea al final del archivo:

STATICFILES\_DIRS =(os.path.join(BASE\_DIR,'static'),)

1. En el archivo base.html de la subcarpeta base agregar lo siguiente:

Antes del TITLE

    <link rel="stylesheet" href="{% static 'css/bootstrap.min.css' %}">

    <title>{% block title %} Document {% endblock %}</title>

Despues del ultimo div cerrado

    {% block footer %}

    {% endblock%}

</div>

<script src="https://code.jquery.com/jquery-1.10.2.min.js"></script>

<script src="{% static 'js/bootstrap.min.js' %}"></script>

Despliegue la pagina : <http://127.0.0.1:8000/inicio/>

**Creacion de Formularios Insertar Registros**

1. Crear en el templates/reservas el archivo reserva\_form.html y digite lo siguiente:

{% extends 'base/base.html' %}

{% block content %}

<form method="post">

    {% csrf\_token %}

    {{ form.as\_p }}

    <button type="submit">Registrar</button>

</form>

{% endblock %}

1. Donde:
   1. {% extends 'base/base.html' %} = va heredar de la plantilla principal
   2. {% csrf\_token %} = token de seguridad que nos pide DJANGO
   3. {{ form.as\_p }} = el formulario lo mostraremos en modo párrafo uno de bajo del otro sin usar posicionamiento especial por ejemplo en un DIV
   4. {% block content %} = todo el formulario que estoy creando va estar metido en el content de la paginaprincipal
2. Dentro de la app “ReservaBarberia” crear el archivo forms.py dentro de ese archivo vamos a importar los forms de DJANGO y digitar lo siguiente:

from django import forms

from ReservaBarberia.models import Reserva

class ReservaForm(forms.ModelForm):

    class Meta:

        model = Reserva

        fields = [

            'codbarbero',

            'codcliente',

            'codservicio',

            'fechareserva',

            'obs',

        ]

        labels = {

            'codbarbero': 'Barbero',

            'codcliente': 'Cliente',

            'codservicio': 'Servicio',

            'fechareserva':'Fecha\_Reserva',

            'obs': 'Obs',

        }

        widgets = {

'codbarbero': forms.Select(attrs={'class':'form-control'}),

'codcliente': forms.Select(attrs={'class':'form-control'}),

'codservicio': forms.CheckboxSelectMultiple(),

'fechareserva': forms.TextInput(attrs={'class':'form-control'}),

'obs': forms.Textarea(attrs={'class':'form-control'}),

        }

1. Ahora modifiquemos nuestro archivo views.vy nuestra aplicación ReservaBarberia que es la que va a recibir la peticion y se va a ejecutar de

from django.shortcuts import render

from django.http import HttpResponse

from ReservaBarberia.forms import ReservaForm

def home (request):

return HttpResponse('Django vive en codigo')

def index(request):

return render(request,'reservas/index.html')

def reserva\_view(request):

if request.method == 'POST':

form = ReservaForm(request.POST)

if form.is\_valid():

form.save()

return render(request, 'reservas/index.html', {'form':form})

else:

form = ReservaForm()

return render(request, 'reservas/reserva\_form.html', {'form':form})

# Create your views here.

1. Ahora modificamos el archivo urls.py de nuestro proyecto principal

from django.contrib import admin

from django.urls import path

from django.conf.urls import url

from ReservaBarberia import views

app\_name = "ReservaBarberia"

urlpatterns = [

path('admin/', admin.site.urls),

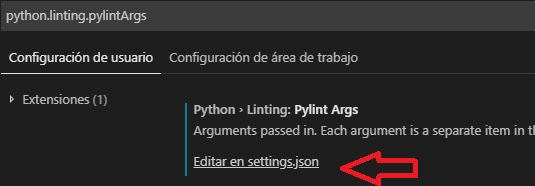
path('',views.home,name='home'),

path('inicio/',views.index,name='index'),

path('nuevo/',views.reserva\_view,name='reserva\_view')

**Listar Registros**

1. Agregar el archivo reserva\_list.html en la subcarpeta reservas de la carpeta templates
2. Actualize el archivo dado en la practica
3. Agregue la siguiente configuracion en Visual Studio code para que haga referencia a los modelos desde la Vista
   1. Menu Archivo/Preferencias/Configuracion
   2. Buscar : python.linting.pylintArgs
   3. Presionar Editar en settings.json pulsar clik:



1. Agregar la siguiente configuracion:

"python.linting.pylintArgs": ["--load-plugins", "pylint\_django"]

1. Agregue la siguiente funcion en la views.py del app reservas

from ReservaBarberia.models import Reserva

def reserva\_list(request):

reserva = Reserva.objects.all()

contexto = {'reservas':reserva}

return render(request, 'reservas/reserva\_list.html', contexto)

1. En el archivo urls.py del proyecto agregue la siguiente urls

path('listar/',views.reserva\_list,name='reserva\_list')

1. Abra su archivo reserva\_lista.html y agregue lo siguiente:

{% extends 'base/base.html' %}

{% block content %}

<table class="table table-bordered">

    <thead>

        <tr>

            <td>#</td>

            <td>Barbero</td>

            <td>Cliente</td>

            <td>Servicio</td>

            <td>Fecha\_Res</td>

            <td>Obs</td>

        </tr>

    </thead>

    <tbody>

{% if reservas %}

{% for reserva in reservas %}

        <tr>

<td>{{ reserva.codreserva }}</td>

<td>{{ reserva.codbarbero }}</td>

<td>{{ reserva.codcliente }}</td>

<td>{{ reserva.codservicio}}</td>

            <td>{{ reserva.fechareserva }}</td>

<td>{{ reserva.obs }}</td>

</tr>

{% endfor %}

{% else %}

<h1>No Se encontraron Reservas</h1>

{% endif %}

    </tbody>

</table>

{% endblock%}

1. Despliegue su aplicación con: <http://127.0.0.1:8000/listar/>
2. Enlace los siguientes hiperviculos desde su archivo base.html

<li><a href="../nuevo">Registrar</a></li>

 <li><a href=../listar>Listar</a></li>

1. Levante la direccion de inicio :<http://127.0.0.1:8000/registrar/>

**Editar Registros**

1. Agregue la siguiente funcion en su archivo View.py

def reserva\_edit(request, codres):

reserva = Reserva.objects.get(codreserva=codres)

if request.method == 'GET':

form = ReservaForm(instance=reserva)

else:

form = ReservaForm(request.POST, instance=reserva)

if form.is\_valid():

form.save()

reservas = Reserva.objects.all()

contexto = {'reservas':reservas}

return render(request, 'reservas/reserva\_list.html', contexto)

return render(request, 'reservas/reserva\_form.html', {'form':form})

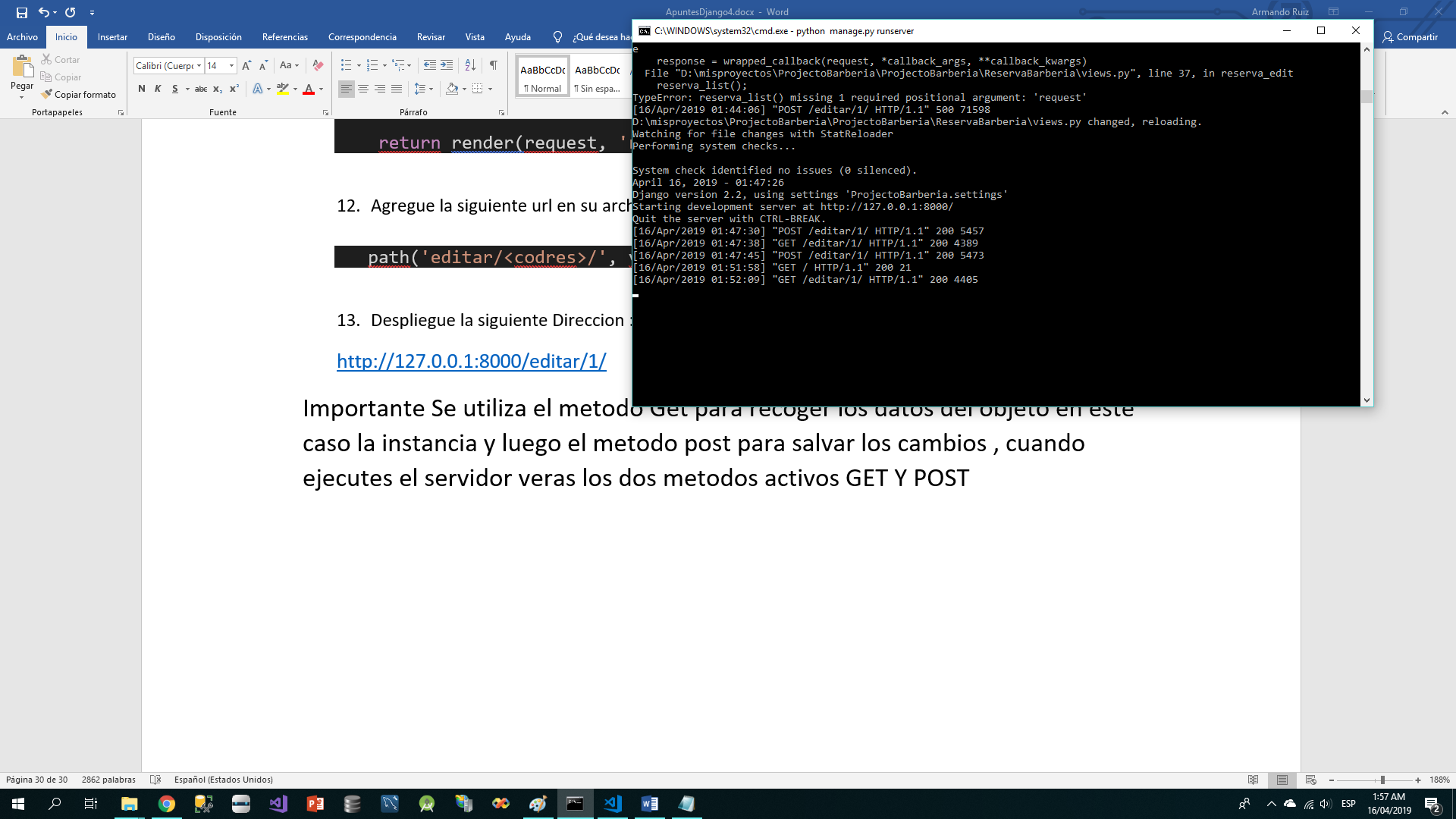
1. Agregue la siguiente url en su archivo URLS.PY:

path('editar/<codres>/', views.reserva\_edit, name='reserva\_editar'),

1. Despliegue la siguiente Direccion : DONDE EL NUMERO uno indica el codigo de reserva a editar

<http://127.0.0.1:8000/editar/1/>

**Importante** Se utiliza el metodo Get para recoger los datos del objeto en este caso la instancia y luego el metodo post para salvar los cambios , cuando ejecutes el servidor y entres a la URL veras los dos metodos activos GET Y POST:



**Agregar el Boton Editar a la Lista de Reservas**

1. Abrir el Archivo reserva\_list.html
2. Agregar la siguiente columna ACCIONES al final

<td>#</td>

            <td>Barbero</td>

            <td>Cliente</td>

            <td>Servicio</td>

            <td>Fecha\_Res</td>

            <td>Obs</td>

            <td>Acciones</td>

1. Agregue al final de la fila lo siguiente:

<td>{{ reserva.codreserva }}</td>

<td>{{ reserva.codbarbero }}</td>

<td>{{ reserva.codcliente }}</td>

<td>{{ reserva.codservicio}}</td>

            <td>{{ reserva.fechareserva }}</td>

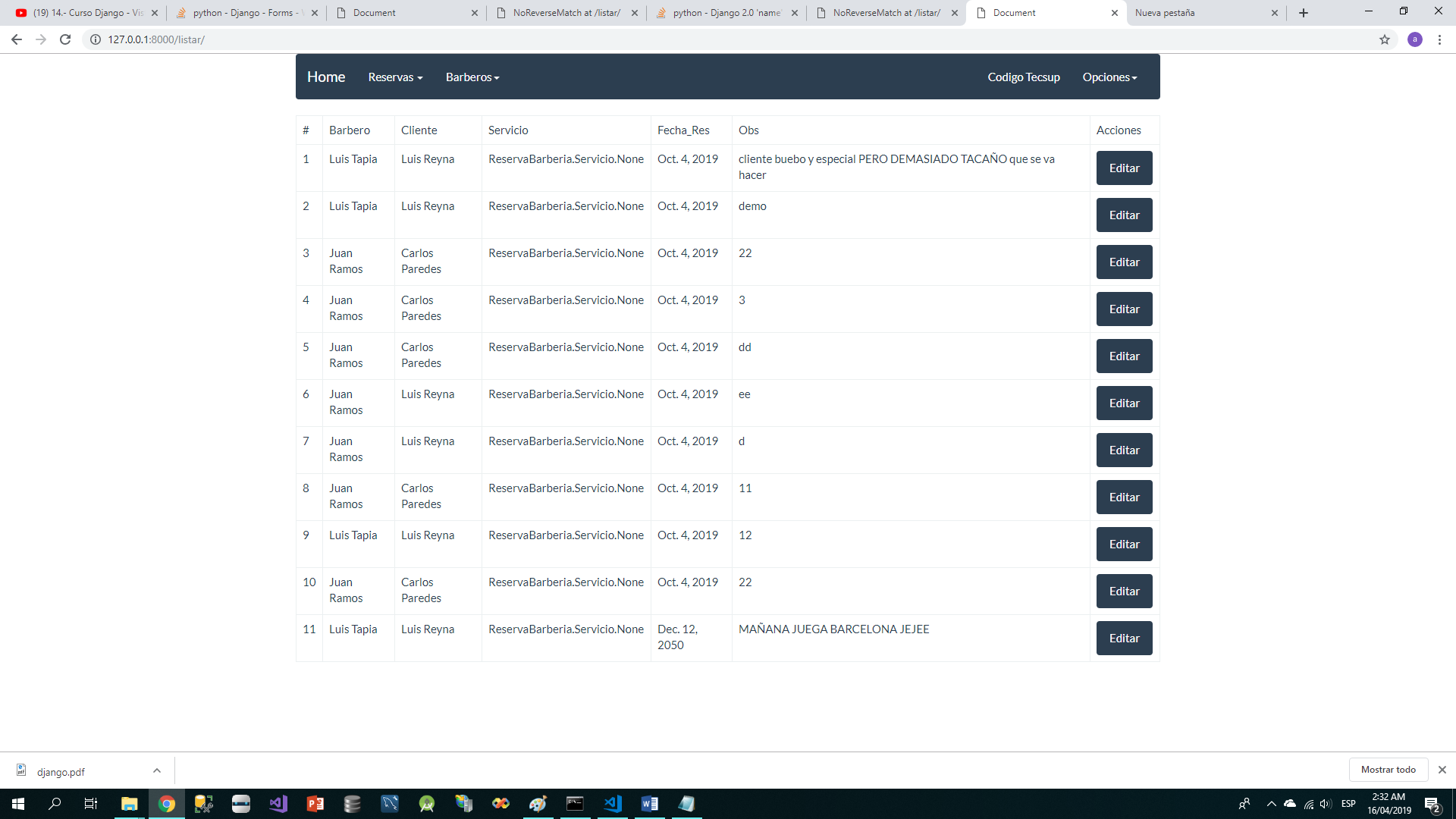
            <td>{{ reserva.obs }}</td>

            <td>

                <a class="btn btn-primary" href="{% url 'reserva\_editar' reserva.codreserva %}">Editar</a>

            </td>

1. Despliegue la aplicación Listar: <http://127.0.0.1:8000/listar/> y vera el boton agregado a la lista



**Eliminar Registros**

1. Agregue la siguiente funcion en su archivo View.py

def reserva\_delete(request, codres):

reserva = Reserva.objects.get(codreserva=codres)

if request.method == 'POST':

reserva.delete()

reservas = Reserva.objects.all()

contexto = {'reservas':reservas}

return render(request, 'reservas/reserva\_list.html', contexto)

return render(request, 'reservas/reserva\_delete.html', {'reserva':reserva})

1. Agregue la siguiente url en su archivo URLS.PY:

path('eliminar/<codres>/', views.reserva\_delete, name='reserva\_delete'),

1. Agregue a su proyecto en la subcarpeta Reservas del proyecto principal el archivo reserva\_delete.html:

{% extends 'base/base.html' %}

{% block content %}

<form method="post">

    {% csrf\_token %}

    <h3>¿Desea eliminar el registro?</h3>

    <button class="btn btn-danger" type="submit">Sí, eliminar</button>

    <a class= "btn btn-default" href="{% url 'reserva\_list' %}">Cancelar</a>

</form>

{% endblock %}

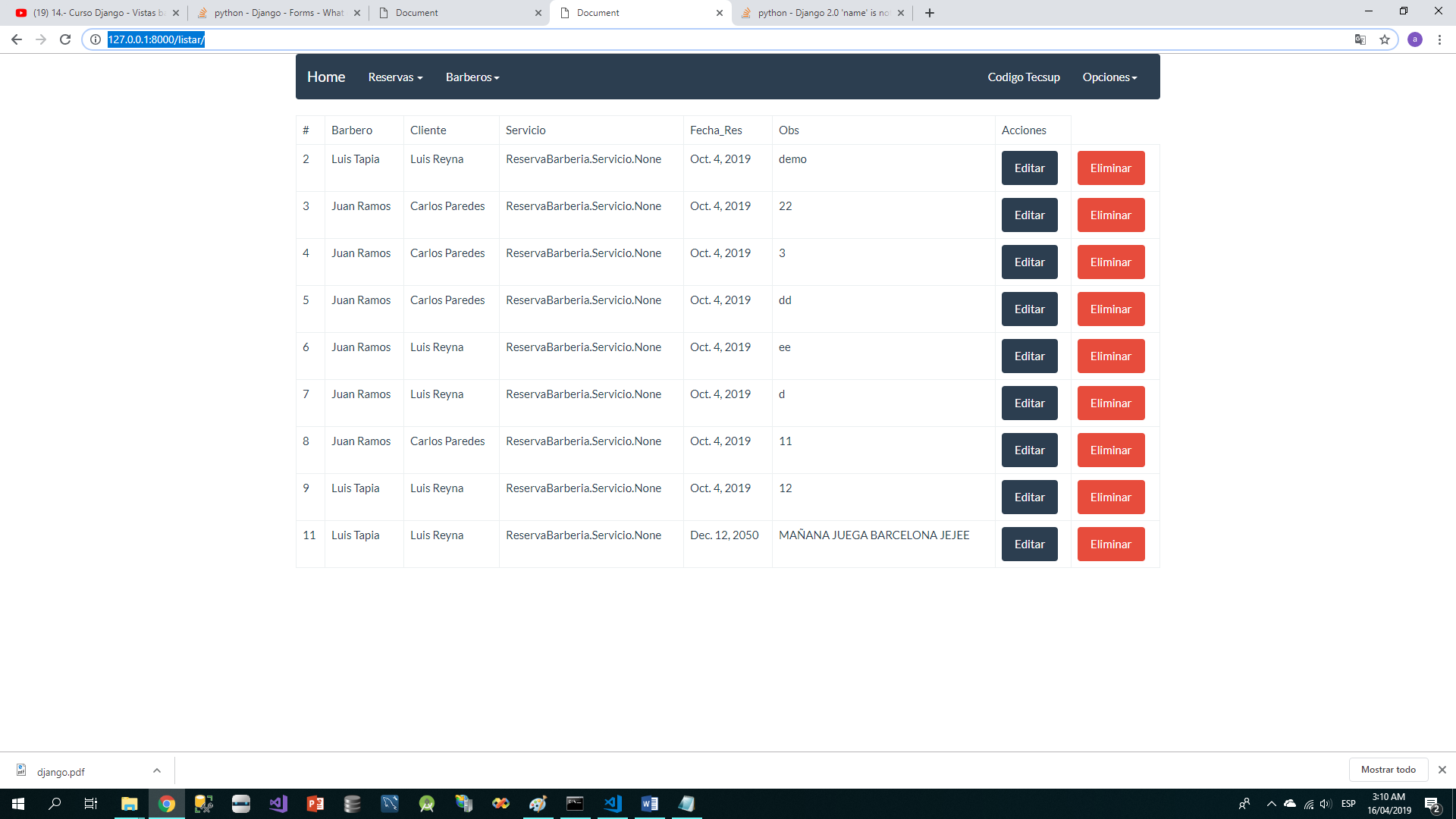
1. Enlacelo en su archivo reserva\_list.html para ejecutarse desde el boton:

  <td>

                <a class="btn btn-danger" href="{% url 'reserva\_delete' reserva.codreserva %}">Eliminar</a>

            </td>

1. Despliegue su URL DE Listar: <http://127.0.0.1:8000/listar/>



Pantalla de Login

1. Agregar el archivo index.html dentro de la carpeta template
2. Dentro del archivo url colocar las siguientes lineas:

from django.contrib.auth.views import LoginView

1. Agregar tanbien el siguiente URL :

path('login/',LoginView.as\_view(template\_name='index.html'),name='login'),

1. En el archivo settings agregar la siguiente configuracion:Al inicio y final

from django.urls import reverse\_lazy

LOGIN\_REDIRECT\_URL = '/listar/'

1. Ejecute la direccion url: <http://127.0.0.1:8000/login/>

**Serializar Objetos Json**

1. Abra el archivo View.py importe la siguiente linea:

from django.core import serializers

1. Agregue la siguiente funcion:

def reservas\_json(request):

lista = serializers.serialize('json',Reserva.objects.all())

return HttpResponse(lista,content\_type='application/json')

1. Abra el archivo URL y agregue el siguiente URL:

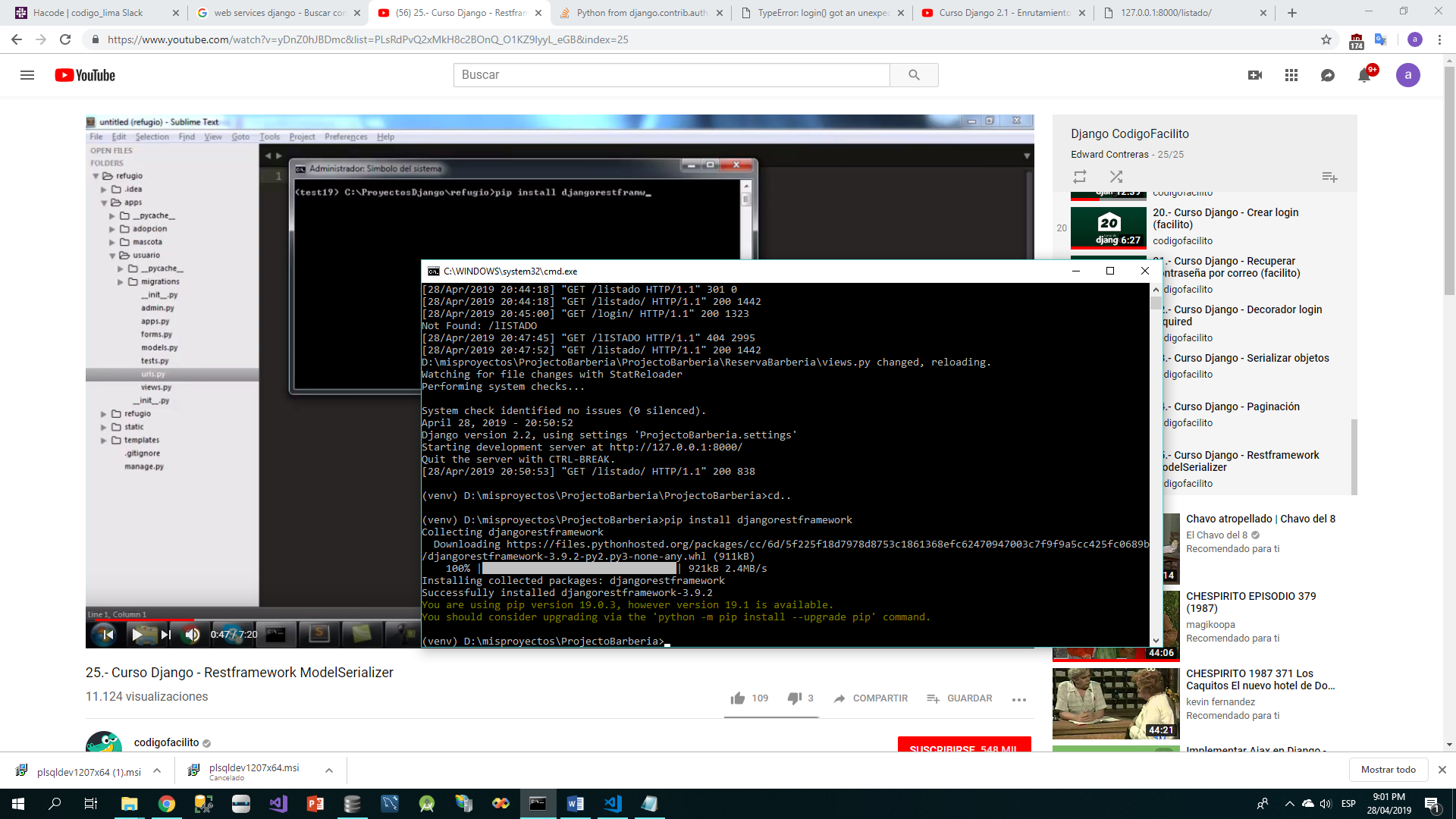
url('listado/',views.reservas\_json,name='listado'),

1. Abra la siguiente direccion : <http://127.0.0.1:8000/listado/>
2. Si modifica la funcion de la siguiente forma solo le mostrara los campos especificados como argumento:

lista = serializers.serialize('json',Reserva.objects.all(),fields=['codbarbero','codcliente'])

**Django RestFramework**

1. Instalar desde tu entorno virtual django restframewok
2. pip install djangorestframework



1. Dentro de la carpeta ReservaBarberia crear el archivo serializer.py con el siguiente contenido:

from rest\_framework import serializers, viewsets

from ReservaBarberia.models import Barbero

class BarberoSerializer(serializers.HyperlinkedModelSerializer):

    class Meta:

        model = Barbero

        fields = ('barberonom','fecharegistro','descripcion')

1. En el archivo views.py agregar el siguiente codigo:

from django.core import serializers

from .serializer import BarberoSerializer

from ReservaBarberia.models import Barbero

from rest\_framework import viewsets

class BarberosList(viewsets.ModelViewSet):

queryset=Barbero.objects.all()

serializer\_class =BarberoSerializer

1. Dentro de la carpeta ReservaBarberia agregar el archivo urls.py

from django.urls import path,include

from .import views

from rest\_framework import routers

router= routers.DefaultRouter()

router.register('ReservaBarberia',views.BarberosList)

urlpatterns = [

path('', include(router.urls)),

]

1. En la raiz principal proyectobarberia en el archivo url agregar lo siguiente:

from rest\_framework import routers

from django.conf.urls import url, include

url('barberos/',include('ReservaBarberia.urls')),

1. En el archivo settings.py agregar:

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

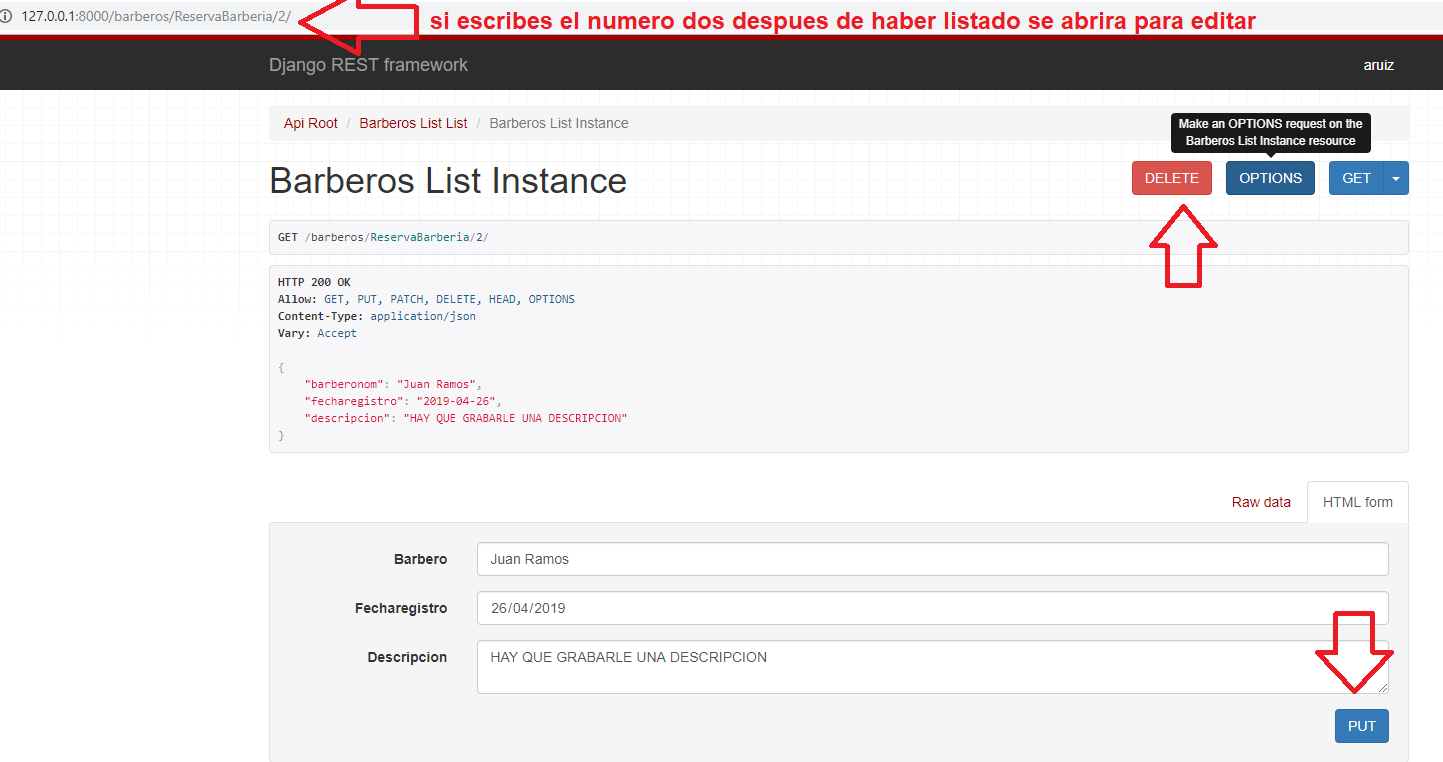
'ReservaBarberia',

'rest\_framework',

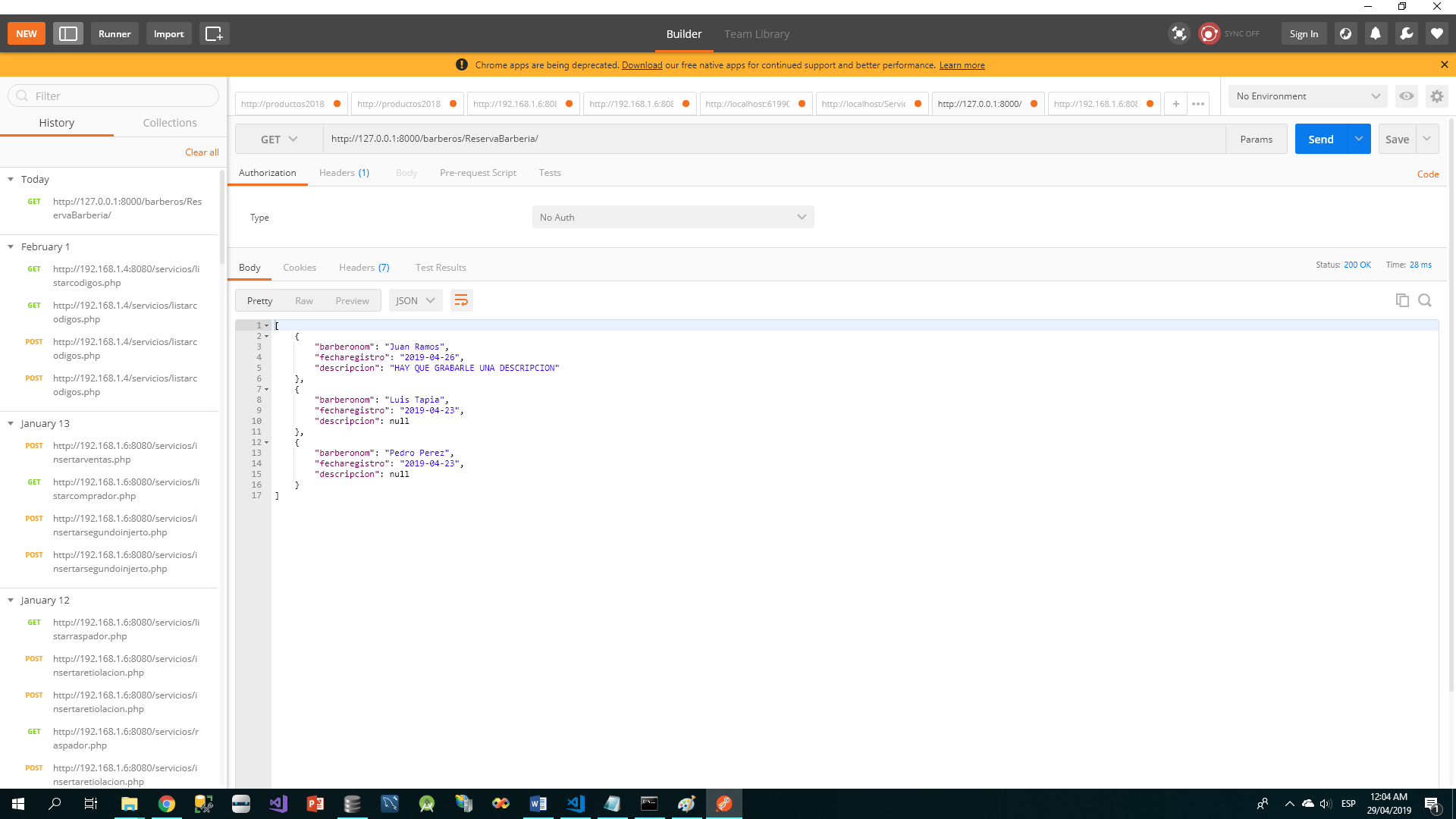
]

1. Probar el rest Api:<http://127.0.0.1:8000/barberos/>

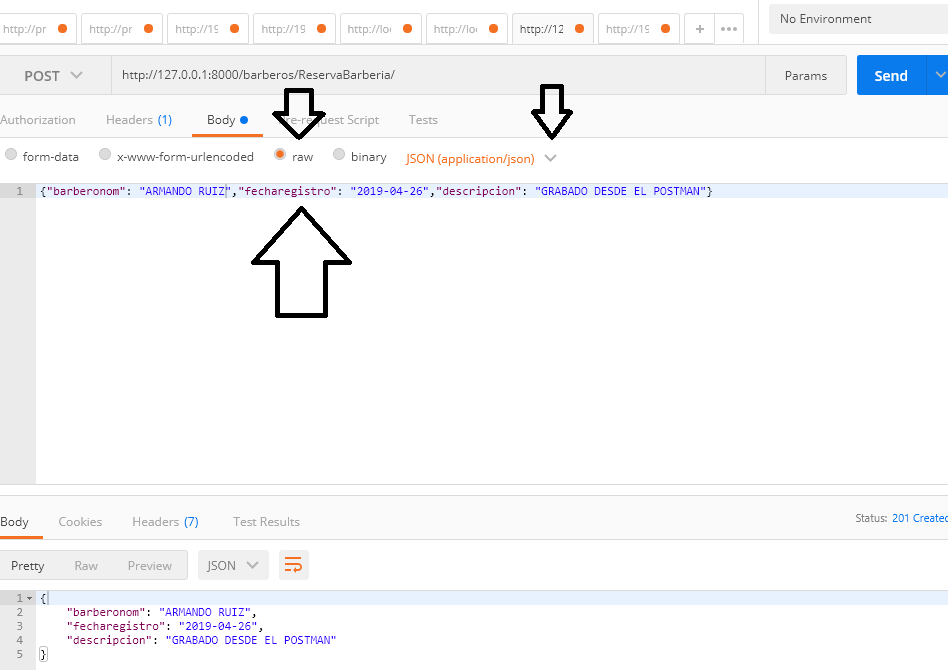




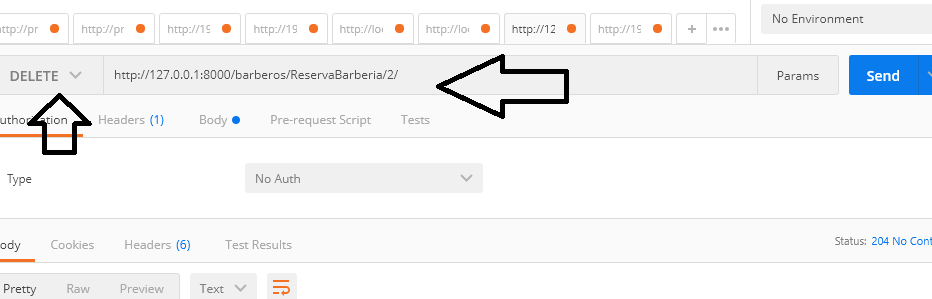
En el postman: Listar



Adicionar



Eliminar



Actualizar:

