## línea horizontal

|  |
| --- |
|  |
|  |

Seminario 3 DDSI

20/12/2020

**─**

Víctor J. Rubia López

Pablo M. Moreno Mancebo

Antonio Galera Gazquez

Juan Andrés Peña Maldonado

Grupo 3

# Parte 1

|  |  |
| --- | --- |
| **NOMBRE ASIGNATURA** | **TÉRMINOS** |
| [Diseño y Desarrollo de Sistemas de Información](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/comunes/diseaoydesarrollodesistemasdeinformacion/!) | SQL |
| [Fundamentos de Redes](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/comunes/fundamentosderedes/!) | HTML |
| [Aprendizaje Automático](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/computacionysistemasinteligentes/aprendizajeautomatico/!) | Python |
| [Ingeniería del Conocimiento](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/computacionysistemasinteligentes/ingenieriadelconocimiento/!) | Python |
| [Modelos Avanzados de Computación](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/computacionysistemasinteligentes/modelosavanzadosdecomputacion/!) | Python |
| [Desarrollo de Sistemas Distribuidos](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/ingenieriadelsoftware/desarrollodesistemasdistribuidos/!) | Python |
| [Desarrollo de Software](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/ingenieriadelsoftware/desarrollodesoftware/!) | Java |
| [Sistemas de Información Basados en Web](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/ingenieriadelsoftware/sistemasdeinformacionbasadosenweb) | HTML, CSS, JavaScript, PHP, JavaServer, XML, AJAX |
| [Administración de Bases de Datos](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/sistemasdeinformacion/administraciondebasesdedatos/!) | SQL |
| [Ingeniería de Sistemas de Información](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/sistemasdeinformacion/ingenieriadesistemasdeinformacion/!) | SQL |
| [Programación Web](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/sistemasdeinformacion/programacionweb/!) | AJAX,CSS,JAVASCRIPT,PHP,XML,DOM,JDOM,JAXP |
| [Sistemas de Información para Empresas](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/sistemasdeinformacion/sistemasdeinformacionparaempresas/!) | XML, Java EE, MySQL, PHP, Python |
| [Sistemas Multimedia](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/tecnologiasdelainformacion/sistemasmultimedia/!) | Java |
| [Tecnologías Web](https://grados.ugr.es/informatica/pages/infoacademica/guias_docentes/curso_actual/tercero/tecnologiasdelainformacion/tecnologiasweb/!) | HTML, CSS, PHP, JavaScript |

# Parte 2

Para esta parte vamos a usar python como lenguaje de programación con flask y como base de datos mongodb, aparte de la librería request de python.

**Requests** es una biblioteca HTTP con licencia de Apache2, escrita en Python. Está diseñado para ser utilizado por humanos para interactuar con el idioma. Esto significa que no tiene que agregar manualmente cadenas de consulta a las URL ni codificar el formulario de sus datos POST.

En los últimos años, **REST** (REpresentational State Transfer) (Transferencia de estado representativo) ha surgido como el diseño arquitectónico estándar para servicios web y API web.

Por comodidad usamos en este seminario un entorno python con la versión 3.8

Toda aplicación Flask es una instancia WSGI de la clase Flask. Se importa dicha clase y creamos una instancia llamada app. Para crear dicha instancia, debemos pasar como primer argumento el nombre del módulo o paquete de la aplicación. Para estar seguros de ello, utilizaremos la palabra reservada \_\_name\_\_. Esto es necesario para que Flask sepa, por ejemplo, donde encontrar las plantillas de nuestra aplicación o los ficheros estáticos.

## Procedimiento

Crear el entorno

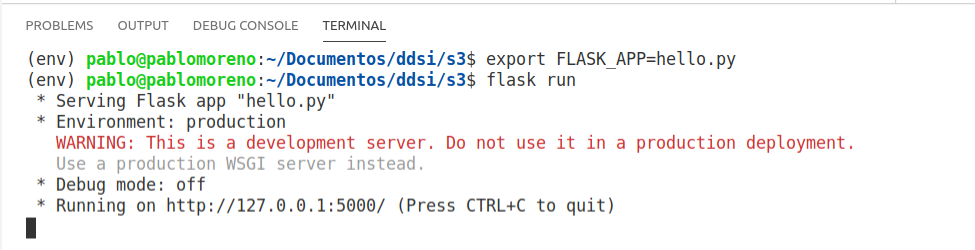
|  |
| --- |
| python3.8 -m venv env  source env/bin/activate |

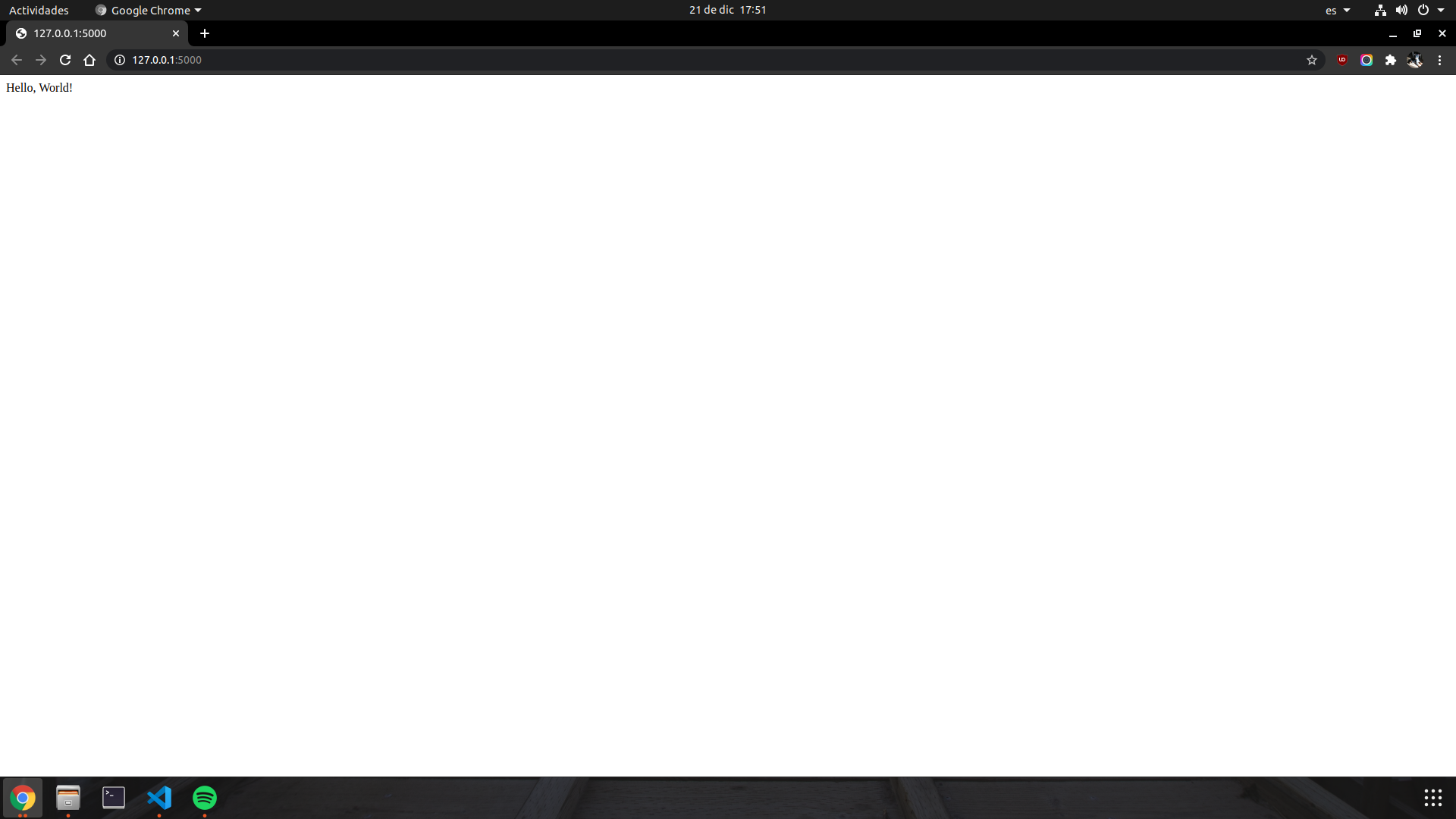
instalar dependencias en el entorno , sacadas con pip freeze

|  |
| --- |
| pip install fast\_json  pip install pymongo  pip install Flask  pip install flake8  pip install requests |

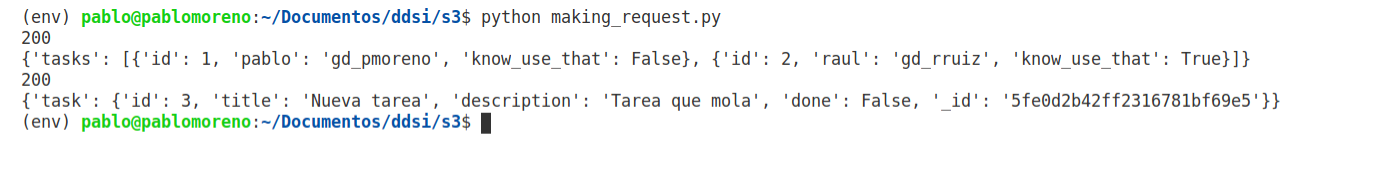
Ejecutar flask

|  |
| --- |
| export FLASK\_APP=hello.py  flask run |

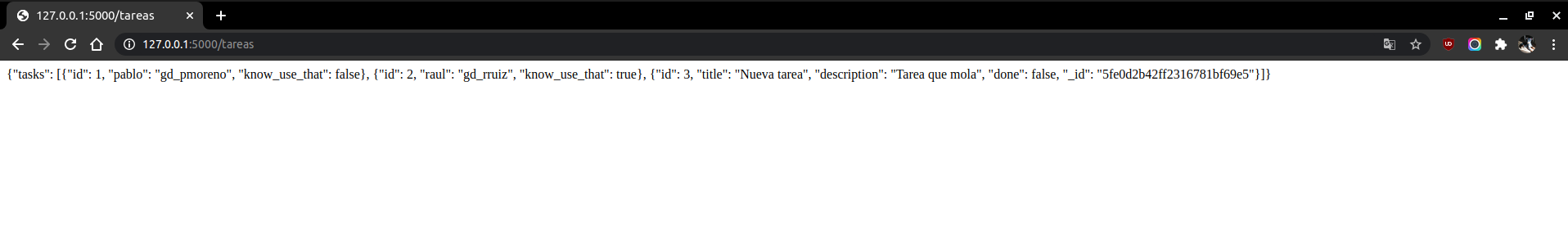




Hacemos la petición



Accedemos a http://127.0.0.1:5000/tareas



## 

## Código del servicio web

|  |
| --- |
| from flask import Flask, request, abort  import complemento\_mongo  import fast\_json  from bson import ObjectId  @fast\_json.convert.register(ObjectId)  def \_id(object\_id):  return str(object\_id)  app = Flask(\_\_name\_\_)  @app.route('/') # Ruta  def hola(): # Funcion que se ejecuta en la ruta de arriba  return "Hello, World!"  # Base de datos en memoria  tareas = [  {  'id': 1,  'pablo': 'gd\_pmoreno',  'know\_use\_that': False  },  {  'id': 2,  'raul': 'gd\_rruiz',  'know\_use\_that': True  }  ]  @app.route("/tareas", methods=['GET'])  def listar\_tareas():  return fast\_json.dumps({"tasks": tareas})  # La respuesta de este metodo no es texto si no un JSON data,  # Con fast\_json lo convertimos a string para imprimirlo en vez del ObjectId(14)  @app.route("/tareas", methods=['POST'])  def nueva\_tarea():  complemento\_mongo.estudiantes.delete\_many({})  if not request.json or 'title' not in request.json:  abort(400)  task = {  'id': tareas[-1]['id'] + 1,  'title': request.json['title'],  'description': request.json.get('description', ""),  'done': False  }  # Conecto a la BD de mongo y lo introduzco  complemento\_mongo.insertDatosNotebook(task)  tareas.append(task)  return fast\_json.dumps({'task': task}), 201  if \_\_name\_\_ == "\_\_main\_\_":  app.run() |

## 

## Mongo

## 

|  |
| --- |
| from pymongo import MongoClient  # Conexión  mongoClient = MongoClient('localhost', 27017)  db = mongoClient.Prueba  estudiantes = db.estudiantes  # Función insertar  def insertDatosNotebook(NBDataconFecha):  estudiantes.insert\_one(NBDataconFecha) |

## 

## Código de hacer una petición

|  |
| --- |
| import requests  # Para obtener una pág web  url = "http://127.0.0.1:5000/tareas"  r = requests.get(url)  print(r.status\_code)  if (r.status\_code == requests.codes.ok):  print(r.json())  h = requests.post(  url, json={'title': 'Nueva tarea', 'description': 'Tarea que mola'})  print(h.status\_code)  print(h.json()) |

# Bibliografía

[Librería Requests](https://www.pythonforbeginners.com/requests/using-requests-in-python)

[Básico de Flask](https://j2logo.com/leccion-1-la-primera-aplicacion-flask/)

[Flask v2](https://blog.miguelgrinberg.com/post/designing-a-restful-api-with-python-and-flask)

[Mongo](https://docs.mongodb.com/manual/tutorial/getting-started/#getting-started)