Desarrollo de APIs con FLASK

Flask es un framework escrito en python que perimte hacer aplicaciones de una manera rápida y con pocas líneas de código. Una API es una interfaz grafica para la comunicación entre la aplicación y software que comparten datos entre ellos. Una API puede se publica que implica que cualquiera puede acceder a la informacíon y tambien puede ser privada, lo que implica que requiere autentificación y en la primera autentificación devuelve un token (Objeto contendeor de datos de autentificación), si el token esta vigente no se pide autentificación y el formato para el token es JWT. REST es una arquitectura de desarrollo de APIs y la parte fundamental para el desarrollo de las misma, las operaciones fuandamentales son los metodos HTTP y son 4 GET (permite leer y consultar información), POST (permite crear nueva información), PUT (permite actualizar la información), DELETE (permite eliminar la información)

Estructura de directorios

```
├──API/
│   └──app./
│      ├──__init__.py
│      ├──routes.py
│
├──test./
│   └──app./
│   │    ├──__init__.py
│   │    ├──routes.py
│   │
│   └──include./
│   └──Lib./
│   └──Scripts./
│   ├──run.py
│   ├──requirements.txt
│
├──run.py
└──requirements.txt
```

Creación de entorno

Instalación del entorno virtual con pip.

```
pip install virtualenv
```

Creación de entorno de pruebas

```
python -m venv test
```

Visión General

Termino	Descripción

Termino	Descripción
app./	Es el paquete en el que reside toda la aplicación Flask.
`__init__.py`	Este fichero contiene métodos para crear e inicializar la app y los distintos componentes y extensiones.
`.gitignore`	Este fichero define los directorios y ficheros que no deben ser tenidos en cuenta por Git.
`README.md`	Fichero en formato markdown en el que indico cosas a tener en cuenta para la ejecución de la aplicación.
`requirements.txt`	Este fichero contiene todas las dependencias Python del proyecto.
`routes.py`	Este fichero contiene toda la lógica de la aplicación: inicialización de la app y las extensiones, vistas, modelos, formularios, etc
`CHANGELOG.md`	Fichero en formato markdown en el que registro las funcionalidades y corrección de errores de cada versión.
`run.py`	Este fichero ejecuta el scripts desde la terminal

Estructura de Archivos

`__init__.py`

```
from flask import Flask, request

app = Flask(__name__)
from app import routes
```

`routes.py`.

```
from flask import Flask, request, jsonify
from app import app

@app.route('/test', methods=['GET'])
def test():
    return jsonify({"respuesta":"Prueba Exitosa"})
```

`run.py`

```
from app import app

if __name__ == "__main__":
    app.run(debug=True,port=XXXX)
```

```
# En producción app.run(host=, port=), donde host, se le indica la dirección que
tendre y port, el puerto que se requiera o no este en uso por otra aplicación.
```

`requirements.txt`

```
Flask==1.1.2
#Instalar las dependencias   con pip install -r requirements.txt
```

CLI Docker

```
FROM python:3.7.1

RUN mkdir -p /files/carpeta
RUN chmod -R 777 /files
RUN mkdir /API

WORKDIR /API
COPY . /API

RUN pip3 install --upgrade pip
RUN pip3  install  -r ./requirements.txt
RUN ls -lrt

CMD ["python3", "./run.py"]
```

Especificación de puertos dentro del contenedor en el archivo `run.py`

```
if __name__ == "__main__":
    app.run(host=0.0.0.0, port=XXXX)
```

Configuración para .gitignore

```
# Byte-compilado / optimización / archivos DLL
__pycache__/
*.py[cod]
*$py.class

# Extenciones de C
*.so

# Distribuciones / empaquetado
.Python
build/
develop-eggs/
```

```
dist/
downloads/
eggs/
.eggs/
lib/
lib64/
parts/
sdist/
var/
wheels/
share/python-wheels/
*.egg-info/
.installed.cfg
*.egg
MANIFEST

# PyInstaller
# Por lo general, estos archivos están escritos por un scritp de Python a partir
de una plantilla. Antes de que PyInstaller complie el exe, para inyectar fecha/
otras informaciones dentro de ello.
*.manifest
*.spec

# Logs de Instalación
pip-log.txt
pip-delete-this-directory.txt

# Pruebas Unitarias / reportes de covertura
htmlcov/
.tox/
.nox/
.coverage
.coverage.*
.cache
nosetests.xml
coverage.xml
*.cover
.hypothesis/
.pytest_cache/

# Traducciones
*.mo
*.pot

# Django stuff:
*.log
local_settings.py
db.sqlite3

# Flask stuff:
instance/
.webassets-cache

# Scrapy stuff:
```

```
.scrapy

# Documentación Sphinx
docs/_build/

# PyBuilder
target/

# Jupyter Notebook
.ipynb_checkpoints

# IPython
profile_default/
ipython_config.py

# pyenv
.python-version

# Archivo de horario celery beat
celerybeat.pid
celerybeat-schedule

# Archivos analizados de SageMath
*.sage.py

# Ambientes
.env
.venv
env/
venv/
ENV/
env.bak/
venv.bak/

# Configuración de proyecto Spyder
.spyderproject
.spyproject

# Configuración de proyecto Rope
.ropeproject

# Documentación mkdocs
/site

# mypy
.mypy_cache/
.dmypy.json
dmypy.json

# Pyre type checker
.pyre/

# Archivos de proyecto en IDEs
.idea/
```

```
.vscode/
.history/
rest-client.env.json

# Archivos de configuración de vistas de carpeta
*.DS_Store
Desktop.ini

# Thumbnails
._*
Thumbs.db
```

Enlaces

- Repositorio oficial de software para python PyPi
- Sitio web Flask Project
- Documentación Oficial Flask Documentatión
- Codigo del Proyecto Github
- Entornos Virtuales virutalenv