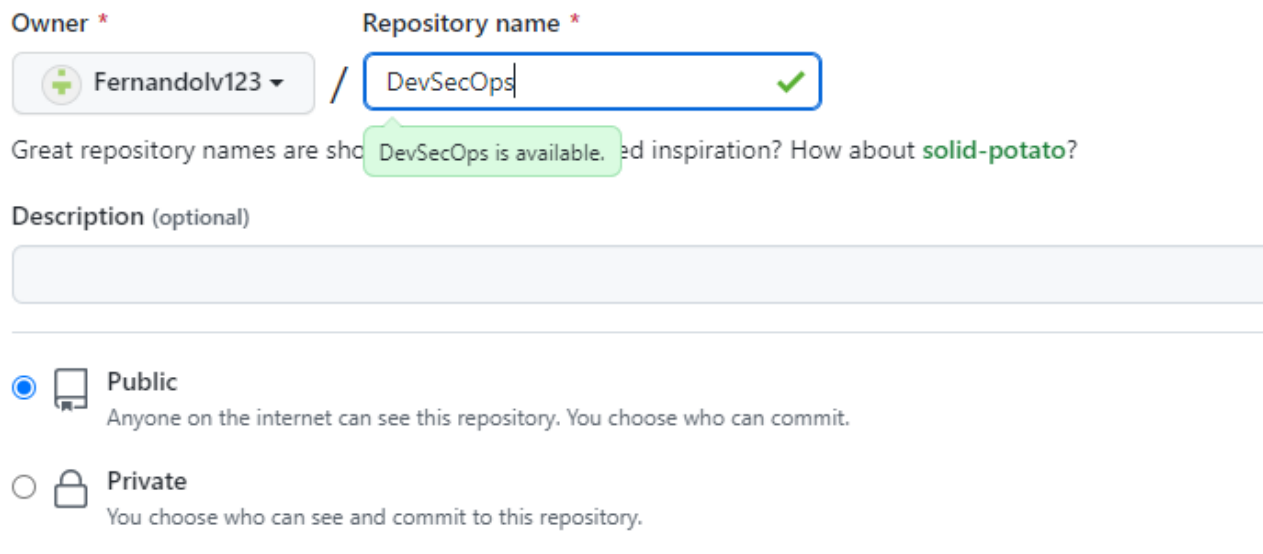


## Preparando GitHub

Primeramente, iremos a GitHub y crearemos un nuevo repositorio público



The screenshot shows the GitHub repository creation interface. The 'Owner' field is set to 'Fernandolv123' with a dropdown arrow. The 'Repository name' field is 'DevSecOps' with a green checkmark. Below the name field, a message says 'Great repository names are short and snappy. DevSecOps is available. Need inspiration? How about solid-potato?'. The 'Description (optional)' field is empty. Under the 'Visibility' section, the 'Public' radio button is selected, with the text 'Anyone on the internet can see this repository. You choose who can commit.' Below it, the 'Private' radio button is unselected, with the text 'You choose who can see and commit to this repository.'

Con el repositorio ya creado, podemos dirigirnos a nuestra máquina ubuntu para subir los archivos.

Una vez en nuestra máquina Linux, podemos utilizar el comando `git config --list` para ver el nombre de usuario y el email que tengamos guardados. Si estos no son los esperados, podemos cambiarlos con los comandos

```
git config --global user.name nombreusuario
```

```
git config --global user.email emailusuario
```

Ahora crearemos una nueva carpeta con el mismo nombre que nuestro repositorio (por comodidad) y ejecutaremos el comando `git init`.

```
usuario@pps:~$ mkdir DevSecOps
usuario@pps:~$ cd DevSecOps/
usuario@pps:~/DevSecOps$ git init
Iniciado repositorio Git vacío en /home/usuario/DevSecOps/.git/
```

Una vez hecho esto, ejecutaremos el comando

```
git remote add origin https://github.com/usuario/proyecto.git
```

```
git remote add origin https://github.com/Fernandolv123/DevSecOps.git en mi caso
```

Con el proyecto git ya preparado, podemos añadir algún fichero de ejemplo. Dado que no he creado el fichero 'README', lo crearé para subirlo.

```
usuario@pps:~/DevSecOps$ echo "Proyecto 2 PPS" > README.txt
usuario@pps:~/DevSecOps$ cat README.txt
Proyecto 2 PPS
```

Una vez creado el fichero, haciendo uso del comando git status, nos lo marcará como archivo sin seguimiento.

```
usuario@pps:~/DevSecOps$ git status
En la rama master

No hay commits todavía

Archivos sin seguimiento:
(usa "git add <archivo>..." para incluirlo a lo que se será confirmado)
README.txt

no hay nada agregado al commit pero hay archivos sin seguimiento presentes (usa "git add" para hacerles seguimiento)
```

Para cambiar esto, es tan fácil como hacer “git add” y volver al comando anterior para comprobar que esté en seguimiento.

```
usuario@pps:~/DevSecOps$ git add README.txt
usuario@pps:~/DevSecOps$ git status
En la rama master

No hay commits todavía

Cambios a ser confirmados:
(usa "git rm --cached <archivo>..." para sacar del área de stage)
nuevos archivos: README.txt
```

Para realizar el commit, utilizaremos el comando “git commit” con el parámetro -m para añadir un comentario.

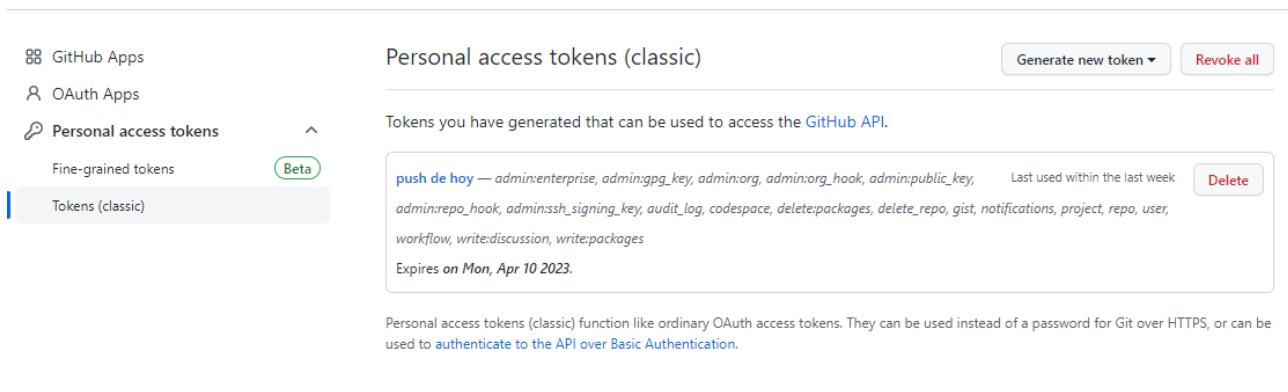
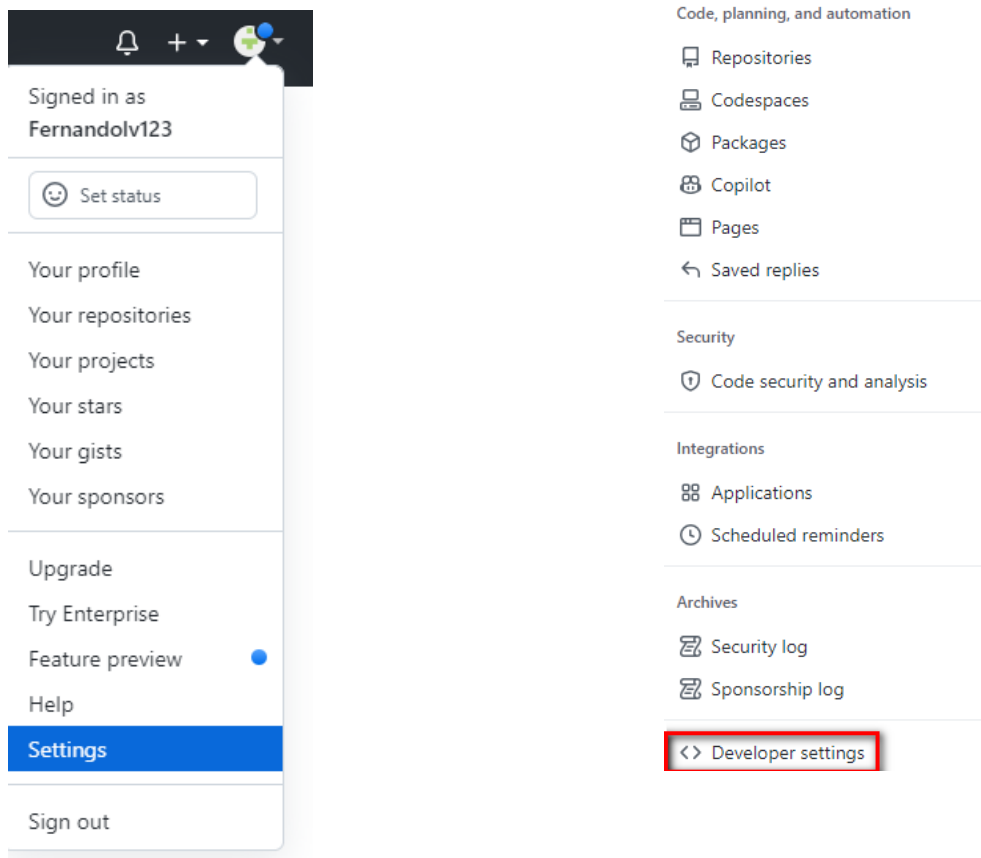
```
usuario@pps:~/DevSecOps$ git commit -m "Fichero README añadido"
[master (commit-raíz) 1420dce] Fichero README añadido
1 file changed, 1 insertion(+)
create mode 100644 README.txt
```

(Si durante este proceso cometemos un error, podemos ayudarnos del comando git reset para sacar el fichero del commit).

Finalmente, tras este proceso, debemos subir nuestro commit a github con “git remote add ” seguido del comando “git push”.

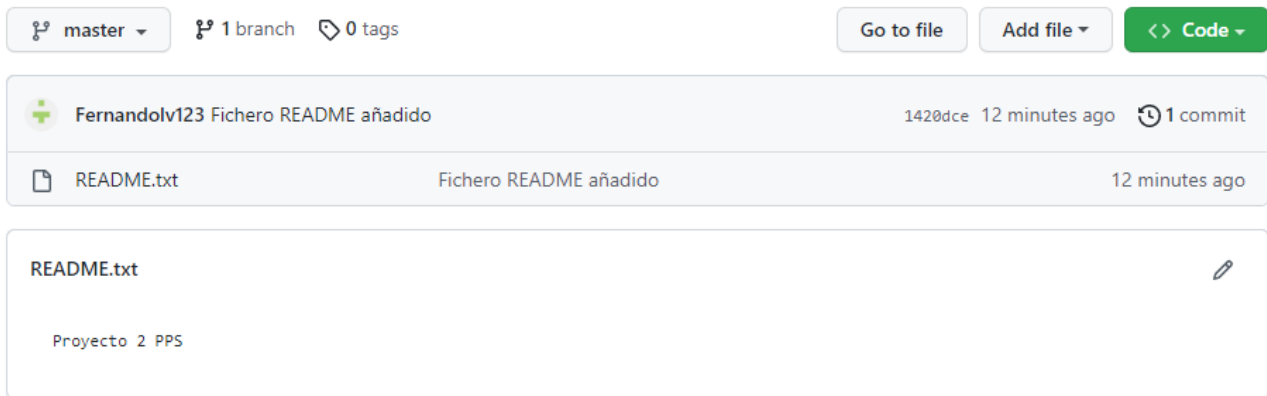
```
usuario@pps:~/DevSecOps$ git remote add DevSecOps http://github.com/Fernandolv123/DevSecOps.git
usuario@pps:~/DevSecOps$ git push DevSecOps
Username for 'https://github.com': Fernandolv123
Password for 'https://Fernandolv123@github.com':
warning: redirigiendo a https://github.com/Fernandolv123/DevSecOps.git/
Enumerando objetos: 3, listo.
Contando objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 256 bytes | 128.00 KiB/s, listo.
Total 3 (delta 0), reusado 0 (delta 0)
To http://github.com/Fernandolv123/DevSecOps.git
 * [new branch]      master -> master
usuario@pps:~/DevSecOps$
```

Debemos utilizar un token en lugar de contraseña por motivos de seguridad. Para generar un token nos dirigiremos a la página de git hub y, sobre nuestro perfil, nos dirigiremos a “Settings → Developer settings → Personal access tokens → Tokens (classic)”



Cuando creamos un nuevo token, podemos especificarle los permisos deseados.

Finalmente, si nos dirigimos a nuestro repositorio en GitHub, podremos ver nuestro fichero añadido



Docker

PHP

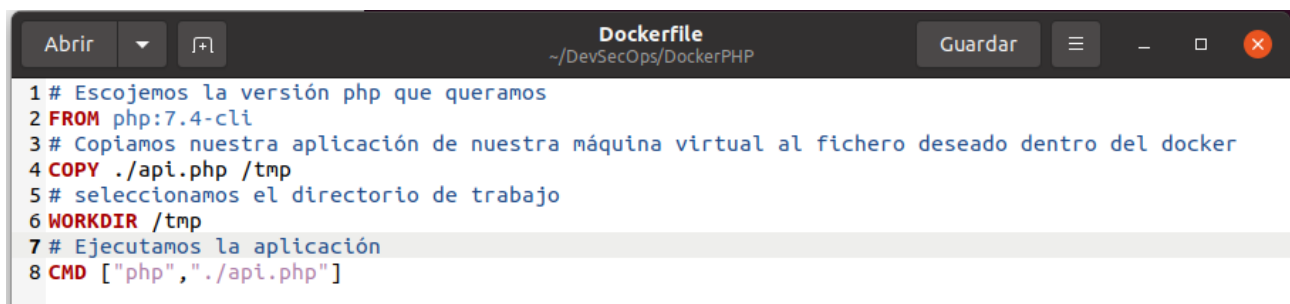
Primero de todo, crearemos una nueva carpeta

```
usuario@pps:~/DevSecOps$ ls
README.txt
usuario@pps:~/DevSecOps$ mkdir DockerPHP
usuario@pps:~/DevSecOps$ cd DockerPHP/
usuario@pps:~/DevSecOps/DockerPHP$
```

creamos una simple aplicación “Hello World” en php



y el fichero Dockerfile



Una vez tenemos ambas cosas, ejecutaremos el comando docker build para construir el docker a partir de la imagen

```
usuario@pps:~/DevSecOps/DockerPHP$ docker build -t dockerphp .
Sending build context to Docker daemon 3.072kB
Step 1/4 : FROM php:7.4-cli
7.4-cli: Pulling from library/php
a603fa5e3b41: Pull complete
c428f1a49423: Pull complete
156740b07ef8: Pull complete
fb5a4c8af82f: Pull complete
972155ae644b: Pull complete
a8e3b94fe6c1: Pull complete
c77004105467: Pull complete
d3e4898bfd25: Pull complete
30f377be4678: Pull complete
Digest: sha256:620a6b9f4d4feef2210026172570465e9d0c1de79766418d3affd09190a7fda5
Status: Downloaded newer image for php:7.4-cli
---> 7bbbb12d1498
Step 2/4 : COPY ./api.php /tmp
---> 0c213b38447f
Step 3/4 : WORKDIR /tmp
---> Running in 5b73dc2a4210
Removing intermediate container 5b73dc2a4210
---> f030e61aaf5c
Step 4/4 : CMD ["php", "./api.php"]
---> Running in 67545563044b
Removing intermediate container 67545563044b
---> 9cbac5763d3e
Successfully built 9cbac5763d3e
Successfully tagged dockerphp:latest
```

La primera vez que lo ejecutamos, nos descargará la versión necesitada por el Dockerfile.

Finalmente, utilizaremos el comando docker run para crear un contenedor

```
usuario@pps:~/DevSecOps/DockerPHP$ docker run --name contenedorphp7 dockerphp
Hello World
```

podemos utilizar parámetros como -d (para ejecutarlo como demonio) o -ti (para hacerlo interactivo).

Como podemos ver en la captura anterior, nuestra aplicación ha funcionado perfectamente.

Podemos utilizar el comando docker ps -a para ver todos nuestros contenedores creados y podemos eliminarlos con docker rm nombrecontenedor (aunque esto está fuera de la práctica).

Ahora crearé este pdf y haré el commit a GitHub con el mismo proceso seguido en el apartado anterior