Informe PTI

Lab 1

Autores

Marc Cabezas

Antonio Arellano

[**1. Familiarización con el entorno**](#_rvjjn5mmfa5m) **3**

[**2. Caso práctico**](#_d6gar49cyccp) **5**

[2.1 new.py](#_a8y5xu9ggbmd) 5

[2.2 list.py](#_4dyq2vhx5sd8) 8

[**3. Conclusiones**](#_luvqh0u8e7mw) **10**

[3.1 Conclusiones de la familiarización con el entorno](#_1crc3n8q7suf) 10

[3.2 Conclusiones del caso práctico](#_fe1hy6ib6zyt) 10

[3.3 Conclusiones finales](#_ceka4eei4egv) 11

# 1. Familiarización con el entorno

Se ha configurado la máquina, instalando apache2 y CGI para poder ejecutar correctamente los archivos html y py.

Se han ejecutado estos comandos:

*sudo apt-get update*

*sudo apt-get install apache2*

*apachectl -V*

*sudo apt-get install git*

*git clone https://gitlab.fib.upc.edu/pti/pti.git*

*ls*

*cd pti/*

*ls*

*cd p1\_cgi/*

*ls*

*sudo mkdir /var/www/html/p1*

*sudo cp \*.html /var/www/html/p1/*

*sudo a2enmod cgi*

*service apache2 restart*

*sudo cp \*.py /usr/lib/cgi-bin/*

*sudo chmod 055 /usr/lib/cgi-bin/*\*

Una vez se ha tenido el entorno preparado, se han realizado pruebas y comprobaciones con los ficheros de ejemplo. Los ejemplos proporcionados intentan mostrarnos la utilización de aquellos componentes que nos serán útiles en el caso práctico, por lo que los ejemplos en si no son muy funcionales, sino que tienen un carácter más pedagógico (figuras 1,2,3,4).

|  |
| --- |
| Screenshot from 2017-02-19 15:32:16.png |
| Figura 1. Formularios de ejemplo. |

|  |
| --- |
| Screenshot from 2017-02-19 15:34:03.png |
| Figura 2. Ejemplo de mensaje de operación satisfactoria. |

|  |
| --- |
| Screenshot from 2017-02-19 15:36:09.png |
| Figura 3. Flujo de la aplicación con diferentes recursos a los cuales acudir. |

|  |
| --- |
| Screenshot from 2017-02-19 15:44:58.png |
| Figura 4. Ejemplo de mensaje de error cuando se introduce erróneamente un dato. |

# 2. Caso práctico

En esta sección se explicarán las soluciones adoptadas para los problemas propuestos, luego éstas soluciones nos ayudarán a elaborar un conclusión.

## 2.1 new.py

Se nos pedía crear un formulario en el cual pudiésemos añadir un nuevo alquiler de coche a una supuesta lista de alquileres de vehículos, en él debían aparecer campos como el modelo del auto, el tipo de motor de éste, el número de días de alquiler, el número de unidades a alquilar, y la base de descuento aplicada al precio final del alquiler. El front-end de este formulario ya se nos facilitaba a modo de ejemplo (figura 5), hemos optado por utilizarlo y crear el script “new.py” el cual implementa la lógica del formulario. Pudiera ser que en el momento de introducir los datos cometamos errores de tipo, como por ejemplo introducir un número negativo de días que alquilamos un vehículo o 0, unidades o números negativos en el descuento, en estos casos hemos optado por capturar los errores en el script e informar al usuario del error volviendo a generar el formulario y añadiendo un mensaje de error, ésta vez desde el script, así podemos probar también cómo generar código HTML desde Python (figura 6). En caso de que el alquiler se añada correctamente, se ha generado por cgi una vista que nos informa de eso y nos da todos los detalles sobre el alquiler que acabamos de añadir (figura 7). Vamos a mostrar a continuación una serie de capturas, las cuales reflejan las diferentes vistas del ejercicio.

|  |
| --- |
| Screenshot from 2017-02-19 15:55:14.png |
| Figura 5. Formulario de registro de alquiler. |

|  |
| --- |
| Screenshot from 2017-02-19 16:00:28.png |
| Figura 6. Formulario con mensaje de error. |

|  |
| --- |
| Screenshot from 2017-02-19 15:55:30.png |
| Figura 7. Vista que se obtiene cuando se ha añadido correctamente un alquiler. |

## 2.2 list.py

También se nos pedía implementar una lista con todos los alquileres registrados en el formulario anterior, se nos proporcionaba un formulario de login (figura 8), ya que se supone que esa lista sólo la puede ver el administrador, por lo que si el login es satisfactorio, se nos mostrará la lista con los alquileres registrados (figura 9). Hemos utilizado el formulario que se nos proporcionaba, y creado el script “list.py” el cual implementa tanto la lógica detrás del formulario como la lógica de mostrar la lista, e igual que en el punto 2.1, si se producía algún error en la introducción de los datos en el formulario, se vuelve a generar el formulario con Python más un mensaje de error (figura 10). Como ya se ha dicho, “list.py” nos genera la lista en HTML desde Python, haciendo uso otra vez de cgi. Vamos a mostrar a continuación una serie de capturas, las cuales reflejan las diferentes vistas del ejercicio.

|  |
| --- |
| Screenshot from 2017-02-19 16:03:46.png |
| Figura 8. Formulario de inicio de sesión. |

|  |
| --- |
| Screenshot from 2017-02-19 15:56:24.png |
| Figura 9. Listado de alquileres registrados. |

|  |
| --- |
| Screenshot from 2017-02-19 15:55:58.png |
| Figura 10. Formulario con mensaje de error. |

# 3. Conclusiones

Las conclusiones la vamos a estructurar de la siguiente manera, primero comentaremos las conclusiones que hemos sacado del punto 1 (Familiarización con el entorno), luego las del punto 2 (Caso práctico) y finalmente a partir de la conclusiones anteriores, intentaremos sacar una conclusión definitiva de las tecnologías con las que hemos trabajado, así como punto a favor y en contra o posibles escenarios en donde éstas tecnologías serían de gran utilidad.

## 3.1 Conclusiones de la familiarización con el entorno

No ha sido muy difícil adaptarse ni hacer el deploy del entorno. La instalación de los paquetes no ha dado excesivos problemas, ni tampoco la prueba de los ejemplos. Tal vez se destacaría la estructura de carpetas descentralizada, ya que los scripts van a parar a un directorio y los HTML a otro, esto quizás en según qué situaciones podría ser poco conveniente.

## 3.2 Conclusiones del caso práctico

No ha sido difícil entender los lenguajes utilizados para el caso práctico, ni tampoco excesivamente complicado entender cómo generar código HTML desde Python, los problemas en esta parte han sido más estructurales que no de código, por ejemplo, hemos tenido problemas con los permisos que se le han de dar a los scripts para que sean ejecutados, otro problema que hemos tenido ha sido con la localización y permisos de ficheros auxiliares, utilizabamos un fichero csv como base de datos en la cual se almacena la lista, bien pues en un principio no sabíamos dónde colocar este fichero para que fuese accedido y luego también tuvimos problemas con los permisos que le teníamos que dar con tal de que los cambios fueran efectivos. Así que en resumen, los problemas encontrados han sido a raíz del sistema de ficheros que implementa el sistema operativo y de la jerarquía de carpetas que utiliza el módulo cgi.

## 3.3 Conclusiones finales

Basándonos en las conclusiones anteriores, éstas tecnologías no tiene un deployment complejo, por lo que podría ser una solución óptima y eficiente para cuando se requiera rapidez y poca complejidad, pudiendo incluso ser una solución para proyectos grandes. Bien es cierto que como se ha visto, se han tenido problemas con el sistema operativo, por lo que sería conveniente entender correctamente cómo interactúan éstas tecnologías con el sistema operativo con el fin de que esos problemas con los permisos no creen una falla de seguridad a la hora de solucionarlos burdamente. Por lo que se puede concluir que la fusión de éstas dos tecnologías, servidor web y cgi, conforman una herramienta con unas posibilidades bastante interesantes y con explotaciones muy diversas. En el campo de la seguridad son tecnologías que no presentan per se fallas de seguridad dramáticas, aunque es cierto que eso dependa más del administrador de sistemas que instale estos módulos y de cómo los configure.