UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Monitor en tiempo real de un sistema de fabricación aditiva para OctoPrint

Presentado por David Zotes González
en Universidad de Burgos — 13 de febrero de 2019
Tutor: César Represa Pérez

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. César Represa Pérez, profesor del departamento de Ingeniería Electromecánica, área de Tecnología Electrónica.

Expone:

Que el alumno D. David Zotes González, [redacted] ha realizado el Trabajo final de Grado en Ingeniería Informática titulado *Monitor en tiempo real de un sistema de fabricación aditiva para Octoprint.*

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 13 de febrero de 2019

Vº. Bº. del Tutor:

D. César Represa Pérez

Resumen

Le tecnología de la impresión 3D está aumentando su presencia tanto a nivel industrial, permitiendo la fabricación tanto de piezas como de prototipos de manera local, barata y rápida, como a nivel doméstico, dotando de capacidades hasta ahora industriales a cualquier persona desde su casa con acceso a esta tecnología.

Debido a la necesidad de crear, usar y producir prototipos en el mundo industrial, la impresión 3D tiene cada vez más presencia en nuestras vidas.

Para facilitar el acceso a la impresión 3D, existen distintas herramientas como OctoPrint. Se trata de una aplicación de software libre que permite modificar los parámetros más importantes de la impresión, tales como, la velocidad de impresión, el flujo de plástico, temperatura de cama y extrusor, etc.

A pesar de la accesibilidad que nos proporciona OctoPrint, debemos tener en cuenta que su principal limitación es que ha sido diseñado para un uso monomáquina. A nivel industrial esto supone un gran problema, ya que se nos presenta la problemática de gestionar múltiples impresoras 3D en una misma plataforma.

Debido a esto, surgió la idea de crear una aplicación web desde la que podamos monitorizar en tiempo real el estado (imprimiendo, operativo, pausado, error), las temperaturas tanto de cama como del extrusor, la pieza actual que esta imprimiendo de varias máquinas de una manera simple e intuitiva. Además, la aplicación web cuenta con un sistema de control de usuarios, el cual permite controlar el acceso a la misma e identificar a los distintos usuarios, cada uno con sus permisos y acciones disponibles de manera que se pueda llevar a cabo un control y una gestión de los recursos de forma segura.

Descriptores

Impresión 3D, OctoPrint, Python, aplicación web

Abstract

3D printing technology is increasing its presence both at industrial level, allowing the manufacture of both pieces and prototypes locally, cheaply and quickly, and at domestic level, providing capabilities to local users that until now where exclusive to industrial level.

Due to the need to create, use and produce prototypes in the industrial world, 3D printing has more and more presence in our lives.

To facilitate this, there are different tools such as OctoPrint. It is a free software project that allows you to modify the most important parameters of printing, such as printing speed, plastic flow, bed temperature and extruder, etc..

Despite the accessibility provided by OctoPrint, we must take into account that its main limitation is that it has been designed for monomachine use. At an industrial level this is a big problem, as we have the problem of managing multiple 3D printers on the same platform.

Due to this, the idea of creating a web application from which we can monitor in real time the state (temperatures of both bed and extruder, the current piece that is printing, etc..) of several machines in a simple and intuitive way came up. In addition, the web application has a user control system,which allows to have control over the access to it and identify the different users, each of them with different permissions and a set of available actions, so that you can carry out a control and management of resources safely.

Keywords

3D Printing, OctoPrint, Python, Web Application

Índice general

Índice de figuras

Introducción

No cabe duda que la impresión 3D cada vez cuenta con más presencia en nuestras vidas ya que permite la creación y producción de prototipos tanto para un entorno doméstico como para un nivel más industrial y todo esto de una manera barata, rápida y sencilla.

La impresión 3D es una solución a la fabricación de prototipos sencilla y polivalente ya que en el mercado encontramos infinidad de materiales que se adecuan a los diferentes prototipos que podamos necesitar. Desde una impresora 3D se pueden imprimir piezas de metal, materiales flexibles e incluso fibra de carbono.

Para entender en que consiste nuestra aplicación web, debemos empezar explicando en que se basa OctoPrint ya que es la base sobre la que se construye nuestro proyecto.

OctroPrint es un sistema que código abierto que sirve para monitorizar y modificar, la mayor parte los parámetros de una impresora 3D, sobre una página web. Pero tiene una limitación importante y es que OctoPrint está diseñado para un uso monomáquina.

Debido a estas carencias, surgió la idea de crear una aplicación web desde la que se pueda controlar, en tiempo real, el estado, la temperatura tanto de la cama como del extrusor, la impresión en curso, el tiempo restante de impresión de varias impresoras 3D.

Con el fin de justificar algunos aspectos del diseño de la aplicación web es necesario que queden claros algunos conceptos clave de la impresión 3D tales como:

- Extrusor: es la pieza que se encarga de arrastrar el filamento, fundirlo y aplicarlo sobre la superficie. Esta compuesto por:

- Motor: es el que se encarga de empujar el plástico hacia el extrusor.
- Rodamiento de presión: es el que se encarga de hacer presión sobre el filamento para que entre de forma continua.
- HotEnd: es la pieza encargada de fundir el filamento para que salga por la boquilla.
- Sensor de temperatura: que se encarga de devolvernos la temperatura a la que se funde el filamento.
- Boquilla de salida: es la pieza que aplica el filamento sobre la superficie de la cama. Existes numerosos tamaños, pero el más extendido es 0.4 mm.

Figura 11: Ejemplo de un extrusor completo. Imagen extraída de [1].

- Cama: es la pieza sobre la que se fija el filamento, normalmente suele ser un cristal. Dependiendo de la máquina, la cama puede tener unas resistencias para que se caliente. Esto es así, porque existen algunos filamentos que necesitan de esta cualidad para que el plástico se adhiera al cristal.

Figura 12: Ejemplo de una cama caliente. Imagen extraída de [5].

- G-code: es el archivo que contiene todas las posiciones de los ejes X, Y, Z que tiene que llevar a cabo la máquina para hacer una impresión completa.

```
638 G02 X33.171944 Y70.454862 Z-0.050000 I3.146309 J-33.690888
639 G02 X36.600000 Y69.507031 Z-0.050000 I-1.004582 J-10.306423
640 G02 X38.104729 Y68.492243 Z-0.050000 I-2.584564 J-5.455393
641 G02 X39.011809 Y67.286112 Z-0.050000 I-2.828059 J-3.071018
642 G02 X39.331246 Y66.514942 Z-0.050000 I-5.323865 J-2.657010
643 G02 X39.807199 Y64.905525 Z-0.050000 I-22.763254 J-7.606872
644 G02 X40.177928 Y63.253731 Z-0.050000 I-25.737692 J-6.644065
645 G02 X40.767720 Y59.882203 Z-0.050000 I-106.718161 J-20.405906
646 G03 X41.575635 Y54.882646 Z-0.050000 I900.292941 J142.919939
647 G03 X41.596655 Y54.841610 Z-0.050000 I0.074477 J0.012248
648 G01 X41.616504 Y54.840865 Z-0.050000
649 G03 X43.832193 Y56.902860 Z-0.050000 I-237.748889 J257.691322
650 G01 X46.048541 Y58.983077 Z-0.050000
651 G01 X46.413663 Y58.586171 Z-0.050000
652 G02 X46.762135 Y58.185736 Z-0.050000 I-7.064100 J-6.499264
653 G01 X46.767705 Y58.155859 Z-0.050000
654 G02 X46.748493 Y58.124940 Z-0.050000 I-0.068315 J0.021020
655 G02 X44.293003 Y55.813775 Z-0.050000 I-522.960908 J553.158477
656 G03 X41.880186 Y53.489265 Z-0.050000 I82.663840 J-88.218658
657 G03 X41.852150 Y53.396102 Z-0.050000 I0.071683 J-0.072371
658 G02 X41.874633 Y53.286906 Z-0.050000 I-23.517215 J-4.898958
659 G02 X42.036314 Y52.491032 Z-0.050000 I-1269.134869 J-258.238797
660 G01 X42.197709 Y51.694954 Z-0.050000
661 G01 X43.390572 Y51.694954 Z-0.050000
662 G02 X53.746174 Y50.143945 Z-0.050000 I-0.014687 J-35.444133
663 G02 X63.838961 Y45.438448 Z-0.050000 I-11.729341 J-38.334901
664 G02 X71.346855 Y38.826978 Z-0.050000 I-18.670215 J-28.770305
665 G02 X75.133285 Y32.095385 Z-0.050000 I-15.602289 J-13.206781
666 G02 X75.630257 Y30.300412 Z-0.050000 I-16.928676 J-5.653304
667 G02 X75.868172 Y28.776325 Z-0.050000 I-11.951813 J-2.646331
668 G03 X75.951985 Y28.284115 Z-0.050000 I3.239037 J0.298300
669 G03 X76.001090 Y28.245795 Z-0.050000 I0.049105 J0.012303
670 G03 X76.050678 Y28.254190 Z-0.050000 I0.000000 J0.150648
671 G03 X76.726523 Y28.498604 Z-0.050000 I-10.207675 J29.282525
672 G01 X76.726571 Y28.498313 Z-0.050000
```

Figura 13: Ejemplo de un G-code en una impresión 3D. Imagen extraída de [6].

Principalmente nuestra aplicación web está orientada al sector industrial. La idea original es usarlo como un monitor para una granja de impresoras 3D.

1.1. Materiales más utilizados en la impresión 3D

A continuación vamos a enumerar los materiales más importantes que se usan en la impresión 3D.

- ABS: es uno de los materiales más utilizados ya que tiene gran resistencia a los choques. Necesita una temperatura de fusión bastante alta, entorno a 230ºC-260ºC. Además para que la impresión sea satisfactoria nuestra impresora 3D deberá contar con la característica de cama caliente. Es decir, necesitamos que la cama esté a unos 90ºC para que la pieza de adhiera a la cama.

- PLA: es un tipo de plástico biodegradable, al contrario que el ABS. Es muy utilizado ya que es muy sencillo de utilizar y no necesitamos cama caliente para que la impresión sea satisfactoria.

- PET: podemos decir que es una mezcla entre los dos anteriores. Es más resistente a choques que el PLA pero es mucho más sencillo de imprimir que el ABS.
- Materiales Flexibles: son muy parecidos al PLA pero tienen la característica de ser flexibles. El resultado final es muy parecido a las fundas TPU para el móvil.

Figura 14: Ejemplo de impresión con material flexible. Imagen extraída de [11].

1.2. Estructura de la memoria

La memoria se ha estructurado siguiendo los siguientes apartados:

- **Objetivos del proyecto**: este apartado explica de forma precisa y concisa cuales son los objetivos que se persiguen con la realización del proyecto.
- **Conceptos Teóricos**: explicación de los conceptos teóricos principales.
- **Técnicas y herramientas**: descripción breve de las técnicas y herramientas utilizadas en este proyecto.

- **Aspectos relevantes del desarrollo del proyecto**: desarrollo de los aspectos mas importantes del proyecto.
- **Trabajos relacionados**: descripción de algunos proyectos similares a *Monitor en tiempo real de un sistema de fabricación aditiva para OctoPrint*.
- **Conclusiones y líneas de trabajo futuras**: descripción de las posibles líneas de trabajo futuras y conclusiones del proyecto.

1.3. Estructura de los anexos

Los anexos se han estructurado según los siguientes parámetros:

- **Plan de Proyecto Software**: estudio de la viabilidad legal y económica y planificación temporal del proyecto.
- **Especificación de requisitos**: describe los requisitos establecidos al comienzo del desarrollo del proyecto.
- **Especificación de diseño**: recoge la información relacionada con el diseño de clases y paquetes así como el diseño de la interfaz.
- **Manual del programador**: instalación de herramientas, compilación, instalación, entorno de ejecución y pruebas.
- **Manual de usuario**: descripción de los aspectos relevantes para el usuario: requisitos, instalación y manual de uso de la aplicación.

1.4. Contenido del CD

El CD aportado junto a esta memoria está organizado en las siguientes carpetas:

- Memoria: incluye la version final en formato *pdf* de la memoria completa del proyecto.
- Anexos: incluye también la version final en formato *pdf* de los anexos de nuestro proyecto.
- Vídeo: contiene un vídeo demostrativo de la aplicación funcionando en un entorno real para mostrar su funcionamiento.
- Código: incluye la última versión de código de la aplicación. [28]

Objetivos del proyecto

El propósito del proyecto es realizar una interfaz gráfica que permita controlar y monitorizar en tiempo real el funcionamiento de una o varias impresoras 3D.

Además se pretende definir diferentes niveles/perfiles de usuario, cada uno con sus permisos y acciones disponibles de manera que se pueda llevar a cabo un control y una gestión de los recursos de forma segura.

2.1. Objetivos funcionales

- Monitorizar el estado de una granja de impresoras 3D conectadas entre sí.
- Crear un sistema de gestión de usuarios y permisos para garantizar la seguridad de la aplicación.
- Permitir comenzar la impresión de una pieza si previamente hemos pre-cargado un G-code [15].
- Permitir pausar una impresión cuando esté en curso.
- Permitir cancelar una impresión cuando esté en curso.
- Permitir conectar una impresora 3D desde la propia aplicación.
- Permitir desconectar una impresora 3D desde la propia aplicación.

2.2. Objetivos no funcionales

- La aplicación se adaptará dinámicamente a cualquier resolución de pantalla y a cualquier tipo de dispositivo.
- La aplicación será lo suficientemente intuitiva para que cualquier usuario sea capaz de utilizarla sin necesidad de ningún tipo de cualificación técnica.
- La aplicación deberá recargarse cada pocos segundos con el fin de que la información sea lo más fiable posible.
- La aplicación utilizará únicamente la red local para las comunicaciones sin necesitar conexión a Internet.

Conceptos teóricos

A continuación vamos a explicar algunos conceptos que son necesarios para comprender como funciona nuestra aplicación web.

3.1. OctoPrint

OctoPrint [10] es una interfaz web que sirve para controlar y monitorizar todos los aspectos que tienen que ver con las impresoras 3D desde el propio navegador.

OctoPrint está principalmente diseñado para funcionar sobre una Raspberry Pi conectada a una impresora 3D. El funcionamiento es sencillo, OctoPrint crea un servidor local desde el que se lanzan todos los servicios para controlar la máquina.

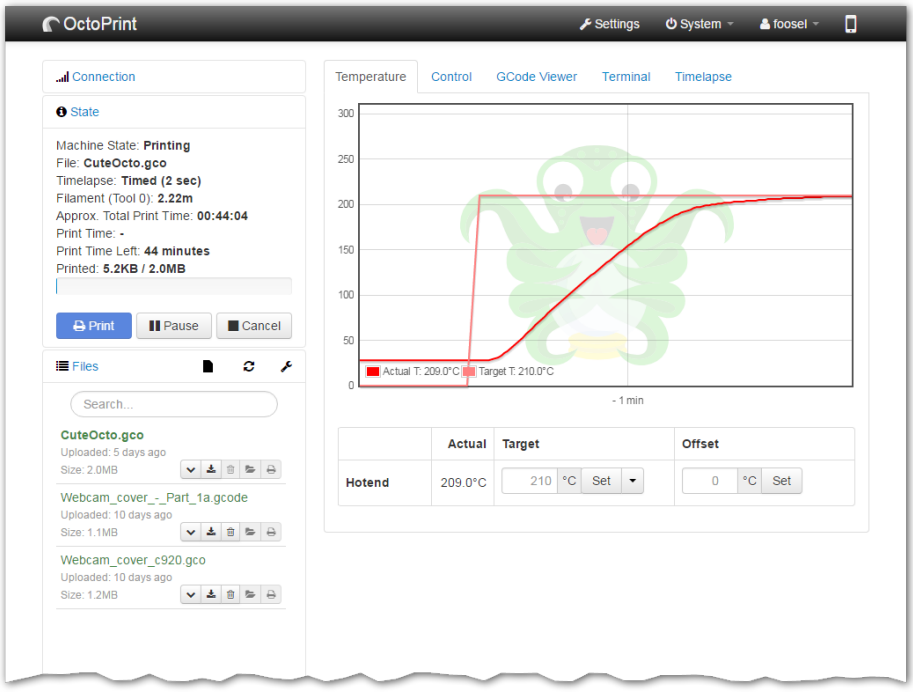

Figura 35: Imagen correspondiente a una instancia de OctoPrint. Imagen extraída de [9].

3.2. API REST

Para entender en que consiste la API [4] de OctoPrint deberemos comprender qué es una API REST.

Una API REST es el conjunto de funciones que los desarrolladores podemos usar para obtener información de una aplicación web. Las solicitudes y las respuestas funcionan a través del protocolo HTTP.

Las operaciones más utilizadas suelen ser GET y POST que son las que hemos usado en nuestra aplicación.

3.3. JSON

JSON [20] es un formato de texto ligero para intercambio de datos.

Es el formato más utilizado cuando llevamos a cabo una comunicación con una API. Como por ejemplo, en nuestro caso cuando hacemos una

petición GET a la API de OctoPrint ésta nos devuelve un archivo JSON en el que se encuentran los datos de las impresoras 3D.


```
{
  "job": {
    "averagePrintTime": 18546.588009119034,
    "estimatedPrintTime": null,
    "filament": null,
    "file": {
      "date": 1517779594,
      "display": "PIH_overhangs_v2_50pc.gcode",
      "name": "PIH_overhangs_v2_50pc.gcode",
      "origin": "local",
      "path": "Tests/PIH_overhangs_v2_50pc.gcode",
      "size": 3679222
    },
    "lastPrintTime": 18546.588009119034
  },
  "progress": {
    "completion": 100,
    "filepos": 3679222,
    "printTime": 18546,
    "printTimeLeft": 0
  },
  "state": "Operational"
}
```


Figura 36: Ejemplo de un archivo JSON en una petición GET. Imagen extraída de [8].

3.4. G-code

G-code [15], se conoce también como RS-274, es el nombre que recibe el lenguaje de programación más usado en control numérico.

Es usado principalmente en automatización y forma parte de la ingeniería asistida por computadora.

En el mundo de la impresión 3D se usa este lenguaje de programación para codificar los archivos que le enviamos a las impresoras 3D. Se podría decir que es un archivo con las coordenadas que debe seguir la impresora para realizar la impresión completa.

Técnicas y herramientas

A continuación vamos a describir las principales técnicas y herramientas que hemos utilizado en nuestro proyecto.

4.1. Técnicas utilizadas para el desarrollo

Python

Python [25] es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.

Para nuestro proyecto hemos elegido Python como lenguaje de programación porque posee una licencia de código libre y además es un lenguaje multiplataforma.

Para ejecutar nuestra aplicación deberemos utilizar una versión de Python 2.7 o superior.

Flask

Flask [14] es un web framework escrito en Python. Está clasificado como microframework porque no necesita otras librerías o herramientas para funcionar.

En nuestro proyecto hemos utilizado Flask ya que nos aportaba numerosas ventajas frente a otras alternativas.

- Usa el motor de plantillas Jinja2 [19] el cual es completamente compatible con Bootstrap [13].

- No se necesita una estructura de un servidor web, ya que Flask crea su propio servidor que se ejecuta en *localhost*
- Tiene un depurador y soporte integrado para pruebas unitarias.
- Flask usa la misma estructura para todos sus proyectos.
- Es compatible con Python3.
- Flask es *open source.*

JavaScript

JavaScript [18] es un lenguaje de programación que se define como orientado objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Generalmente JavaScript se usa en el lado del cliente. Su principal uso es en los navegadores, a la hora de crear páginas web. Con esto podemos hacer web dinámicas y hacer mejoras visuales en las páginas web.

En nuestro caso, hemos utilizado JavaScript para ejecutar las funciones de conectar/desconectar la impresora, pausar, comenzar/cancelar impresión y para introducir confirmación en todos los botones que tengan una función crítica.

HTML

HTML [17] es un lenguaje marcado para la elaboración de páginas web. Actualmente se puede decir que HTML es el lenguaje estándar que se ha impuesto en la visualización de páginas web, ya que es el que todos los navegadores del momento han adoptado.

Hemos utilizado HTML para crear la parte visible de nuestra aplicación web.

En nuestra aplicación hemos creado una base (que contiene en *navbar* y el *footer* de la aplicación), un índice (contiene la información que queremos mostrar de las máquinas) y por último, también contamos con la página del *login*.

SQLite

SQLite [26] es un sistema de gestión de bases de datos relacional.

Tenemos que saber que SQLite no es un motor de base de datos cliente-servidor al uso, sino que está integrado dentro del propio programa.

Sabemos que hay mejores opciones a la hora de crear una base de datos, pero entendimos que nuestro proyecto no se centraba en la base de datos y no necesitábamos una base de datos tan grande, puesto que solo se iba a usar para la gestión de los usuarios. Es por esto que nos decidimos a utilizar SQLite en lugar de otras herramientas como MySQL o PostgreSQL.

Nginx

Nginx [22] es un servidor web/proxy inverso de alto rendimiento y un proxy para protocolos de correo electrónico. Además tiene un licencia de software libre y de código abierto lo cual es perfecto para nuestra aplicación.

En nuestro caso, hemos usado Nginx para que las llamadas a las impresoras desde el navegador fueran con un determinado nombre.

PyCharm

PyCharm [24] es un entorno de desarrollo integrado utilizado en la programación de las computadoras, especialmente en Python.

Hemos utilizado PyCharm como nuestro IDE principal para desarrollar nuestro proyecto, ya que cuenta con la ventaja de ser multiplataforma y tiene soporte integrado para el control de versiones.

Además, gracias a la cuenta de correo de la Universidad de Burgos tenemos acceso a una licencia de uso profesional de esta herramienta.

Bash

Bash [12] es un programa informático, cuya función consiste en interpretar órdenes, y un lenguaje de consola.

Nuestra aplicación se ejecuta sobre un ordenador con Debian 9 instalado, y hemos que tenido que utilizar Bash para hacer una serie de scripts. Estos scripts se encargan de lanzar todas las instancias de OctoPrint y todas las cámaras que controlarán las impresoras.

Sublime Text

Sublime Text [27] es un editor de texto y editor de código fuente escrito en C++ y en Python.

Tomamos la decisión de utilizar este editor ya que es multiplataforma y completamente gratuito.

Sublime Text lo hemos usado principalmente para la creación y edición de los scripts en Bash que hemos comentado anteriormente. Y en alguna que otra ocasión para editar los archivos HTML y CSS con los que cuenta nuestra aplicación web.

PuTTY

PuTTY [23] es un cliente SSH, Telnet, rlogin, y TCP raw con licencia libre. Anteriormente solo estaba disponible en Windows pero actualmente es un cliente multiplataforma.

Nuestro uso principal de esta herramienta ha sido como cliente SSH para controlar el servidor donde corre nuestra aplicación web. De esta manera, podemos lanzar o detener nuestros scripts sin tener que estar en la misma localización que el servidor en el que se ejecuta nuestra aplicación.

4.2. Control de versiones y documentación

A continuación, vamos a explicar la herramienta que hemos usado para el control de versiones y la que hemos decidido utilizar para desarrollar la documentación nuestro proyecto.

GitHub

GitHub [16] es una plataforma de desarrollo colaborativo para alojar proyectos y utiliza el control de versiones llamado Git.

Por defecto todo el código que se aloja en esta página es de dominio público pero si compramos la versión de pago podemos hacer que nuestros repositorios sean privados.

Actualmente, gracias a la cuenta de correo de la Universidad de Burgos podemos tener repositorios privados sin tener que pagar por ello.

En nuestro caso, hemos utilizado GitHub para llevar un control de versiones del proyecto.

LaTeX

LaTeX [21] es un sistema de composición de textos, orientado a la creación de documentos escritos.

Hemos elegido LaTeX porque tiene una calidad tipográfica superior a la de otros editores convencionales.

Para nuestro proyecto hemos utilizado TexMaker en su versión para Mac OS para redactar la memoria y los anexos.

Aspectos relevantes del desarrollo del proyecto

A continuación vamos explicar los detalles principales del desarrollo de nuestro proyecto. Haremos un repaso de todos los pasos importantes que hemos tenido que llevar a cabo para que nuestra aplicación web funcione sin problemas.

5.1. Instalación de las herramientas

Instalación de las instancias de OctoPrint

Para explicar los aspectos más relevantes de nuestro proyecto deberemos empezar comentando qué fue lo primero que hicimos.

En nuestro caso, lo primero que hicimos fue instalar Debian en su versión número 9 en el ordenador que hará la función de servidor.

Para evitar problemas deberemos crear un usuario por cada instancia de OctoPrint que vayamos a instalar; de esta manera podemos diferenciar cada máquina con un usuario.

El siguiente paso que debemos hacer es instalar una instancia de OctoPrint por cada una de las máquinas que queramos tener en nuestra aplicación web. Para ello deberemos clonar el repositorio de GitHub de OctoPrint [7]. Además de clonar el repositorio de GitHub deberemos crear un entorno virtual de Python y ejecutar una serie de comandos que dejo a continuación:

```
1. git clone https://github.com/foosel/OctoPrint
```

```
2. cd OctoPrint

3. virtualenv venv

4. ./venv/bin/pip install pip --upgrade

5. ./venv/bin/python setup.py install

6. mkdir ~/.octoprint
```

Para probar si la instalación ha ido bien, deberemos introducir el siguiente comando:

```
~/OctoPrint/venv/bin/octoprint serve
```

Deberemos repetir esta operación con cada usuario que hayamos creado previamente.

Primeros pasos

La primera vez que arrancamos una instancia de OctoPrint deberemos seguir el asistente de instalación. Es muy sencillo, no tienen ninguna complicación, tan solo deberemos seguir los pasos indicados e indicar el tamaño de la cama caliente de nuestra impresora 3D.

Una vez tengamos bien configurado nuestro OctoPrint deberemos ir al apartado de configuración y guardaremos el Token de la API. Tenemos que tener en cuenta que es una clave diferente para cada máquina.

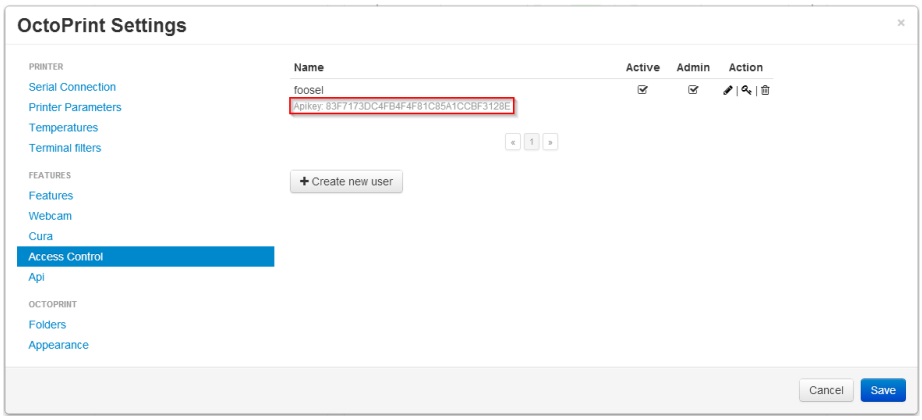

Figura 57: Acceder al Token de la API.

Una vez tengamos el Token anotado podremos comenzar a desarrollar nuestra aplicación.

5.2. Funcionalidad

Leer datos del CSV

Una vez tenemos todos los datos de la impresora 3D tales como puerto, nombre de la máquina, Token de la API procederemos a crear los archivos CSV con todos los datos tal y como se ve en la siguiente ilustración:

```
1  Puerto;Maquina;Token API
2  5001;maq1;50EA1BD3AACC4133B32C907E626C4FA7
3  5002;maq2;56D24EE0B4A64CD2AB9545461152CA96
4  5003;maq3;4B22BE0A43DA4DF7999B228F106260A2
5  5004;maq4;8C5E7DB1649F4069AAE6C1C5013591F5
6  5005;maq5;AA6675CD2AF04C29A42815F2EDA5B8B5
7  5006;maq6;912088BC02A44629AA15A159E61579E4
8  5007;maq7;3DE1E4F69D9E47A6B57D9ECD7086E949
```

Figura 58: Datos de las impresoras en formato CSV.

A continuación deberemos leer los datos de este archivo desde nuestro 'back-end' de la aplicación. Durante la lectura del archivo iremos guardando la información en un diccionario de Python. De esta manera tendremos los datos de todas las máquinas organizados en un diccionario y no tendremos

que editar el código cada vez que queramos añadir una máquina nueva; tan solo tendremos que añadir los datos de la máquina en el archivo CSV.

Peticiones GET a la API

La API que dispone OctoPrint cuenta con numerosos comandos para extraer la información de las impresoras 3D. Dependiendo de la información que necesitemos usaremos unos comandos u otros. Los comandos que hemos usado en la aplicación web son *Printer* y *Job*.

- *Printer*: mediante este comando podemos obtener información sobre los parámetros de la impresora, como estado, temperaturas, etc.
- *Job*: con este otro comando podemos obtener información sobre la pieza actual que se está imprimiendo.

Si la petición ha funcionado correctamente, obtendremos un archivo de tipo JSON con toda la información de la máquina.

Además, en la función que se encarga de realizar las peticiones deberemos incluir una serie de excepciones para que la aplicación siga funcionando aunque alguna impresora 3D nos devuelva un error.

Debemos saber que cuando generamos una petición GET ésta nos devuelve una serie de números que nos indican como ha ido dicha petición. Tipos de respuestas de una petición GET en nuestra aplicación:

- 200: la petición ha salido correctamente.
- 204: la respuesta de la API no tiene contenido.
- 404: no hay comunicación por parte del servidor.
- 409: la impresora no está operativa.

Identificando este tipo de respuestas podemos capturar los posibles errores que se pueden producir en la aplicación. De esta manera si se producen alguno de ellos identificaremos el error pero la aplicación seguirá funcionando sin problemas. Capturando dichas respuestas nuestra aplicación ganará en estabilidad.

Extraer la información útil del archivo JSON

Lo que hacemos a continuación es llamar a otra función que se encarga de leer todos los datos del JSON y filtramos solo los datos y los introducimos en un diccionario de Python. De esta manera tendremos un diccionario de Python con todos los datos que nos interesan, es decir, sólo los que vamos a mostrar en nuestra aplicación.

Una vez tengamos todos los datos en sus sitio, devolveremos el diccionario para poder mostrarlo en nuestra aplicación.

Peticiones POST a la API

El otro tipo de operaciones que usamos en nuestra aplicación son las peticiones POST.

La principal diferencia entre una petición GET y una POST, es que la petición GET aparece en la URL y por tanto es visible para el usuario. Mientras que si hacemos una petición POST ésta no sale en la URL y es transparente al usuario.

Por este motivo, cuando nosotros queremos que la API nos de información lo hacemos con una petición GET, pero cuando queremos dar una orden, como por ejemplo desconectar una impresora 3D usamos una petición POST ya que debemos incluir el Token de la API en el comando y esa información debe permanecer oculta.

A continuación vamos a explicar cómo se realiza la desconexión de una máquina usando una petición POST.

Nuestra petición POST está compuesta principalmente por dos elementos llamados 'data' y 'headers'.

El elemento 'headers' contiene el tipo de archivo que estamos enviando y el Token correspondiente a impresora 3D sobre la que queremos ejecutar la operación.

Y el elemento 'data' contiene el comando que se va a ejecutar en la impresora, en este caso es la operación de desconectar una máquina.

En la figura 59 se muestra un ejemplo de cómo se realiza la desconexión en nuestra aplicación web:

```
1  headers = {'Content-Type': 'application/json','X-Api-Key': maquina[2]}
2  data= '{"command": "disconnect"}'
3  urlConectar= str(host + ":" + maquina[0] + "/api/" + conn)
4  peticion=requests.post(urlConectar,data=data,headers=headers)
```

Figura 59: Desconexión de una máquina en nuestra aplicación web.

Añadir nuevas impresoras

En la última versión de la aplicación hemos mejorado enormemente ésta funcionalidad. En antiguas versiones si queríamos añadir alguna máquina teníamos que hacer la instalación estándar de OctoPrint pero luego teníamos que entrar en el *back-end* de la aplicación y teníamos que añadir en un diccionario de Python todos los datos de configuración tales como Token de la API, puerto de conexión, nombre de la máquina, etc. Luego teníamos que ir al archivo HTML y añadir línea a línea todas las características que queríamos mostrar, ya que la interfaz desarrollada en ese momento mostraba la información de cada máquina por separado, es decir, por cada cada máquina se mostraba en pantalla lo que hemos denominado una tarjeta informativa.

Con el fin de mejorar la escalabilidad de la aplicación hemos incluido el uso de archivos CSV. Al añadir esta funcionalidad sólo tenemos que añadir todos los datos de la nueva impresora en el archivo CSV. Es decir, no tenemos que modificar el código de la aplicación. En el HTML tampoco deberemos modificar nada puesto que ahora solo existe una tarjeta a partir de la cual se dibuja el estado de todas las impresoras.

Scripts

Ésta ha sido una tarea especialmente complicada ya que no tenía mucha experiencia previa tratando con este tipo de tareas.

Principalmente debemos modificar dos *Scripts* que se generan en la instalación de OctoPrint. En el servidor los *Scripts* se encuentran situados en las siguientes rutas */etc/init.d y etc/default.*

Tenemos que tener en cuenta que tenemos un *Script* por cada instancia de OctoPrint por lo que tenemos que duplicar el *Script* tantas veces como máquinas tengamos instaladas en la aplicación. Deberemos modificar todos los *Scripts* creando una variable que dependa del número de máquina. En nuestro caso hemos creado un variable llamada *INSTANCENUMBER* que se incrementa con cada máquina. Una vez tengamos esto, deberemos cambiar

el nombre del paquete añadiendo el nombre de la variable tal y como vemos a continuación:

```
17.      INSTANCENUMBER=1
18.
19.      PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
20.      DESC="OctoPrint $INSTANCENUMBER Daemon"
21.      NAME="OctoPrint $INSTANCENUMBER"
22.      PKGNAME=octoprint$INSTANCENUMBER
23.      PIDFILE=/var/run/$PKGNAME.pid
24.      SCRIPTNAME=/etc/init.d/$PKGNAME
25.      DEFAULTS=/etc/default/$PKGNAME
26.
```

Figura 510: Ejemplo de nuestro *Script*.

Deberemos seguir los mismos pasos para los *Scripts* que se encuentran en las dos rutas mencionadas anteriormente.

Una vez hayamos seguido todos estos pasos deberemos ponernos con el *Script* que se encarga de lanzar nuestra aplicación.

Con la aplicación copiada en la raíz del servidor deberemos copiar los *Scripts* anteriormente citados una vez más, pero esta vez no tendremos la variable *INSTANCENUMBER* sino que deberemos cambiar el nombre del paquete por el que tiene nuestra aplicación, en nuestro caso *monitorOcto* e indicaremos la ruta en la que se encuentra dicho archivo tal y como mostramos a continuación:

```
19.      PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
20.      DESC="Monitor OctoPrint Daemon"
21.      NAME="Monitor OctoPrint"
22.      PKGNAME=monitorOcto
23.      PIDFILE=/var/run/$PKGNAME.pid
24.      SCRIPTNAME=/etc/init.d/$PKGNAME
25.      DEFAULTS=/etc/default/$PKGNAME
26.
```

Figura 511: Ejemplo del *Script* que lanza nuestro monitor.

En el código de nuestra aplicación deberemos asegurarnos que la dirección desde la que pedimos los datos es la misma que tiene el servidor y también tenemos que tener en cuenta el puerto sobre el que queremos que funcione nuestra aplicación, para ello deberemos indicarlo en la configuración de Flask dentro del archivo *monitorOcto.py*.

5.3. Diseño de la interfaz

Vista general de la aplicación

Uno de nuestros objetivos era mostrar una tarjeta por cada máquina que tengamos incluida en la aplicación. Además, ésta tarjeta tiene un color diferente dependiendo del estado en que se encuentre la máquina.

En cada tarjeta mostraremos los datos que para nosotros son interesantes, en nuestro caso son:

- **Estado**: muestra el estado de la máquina en todo momento. Puede ser imprimiendo, operativa o que la máquina tenga algún error.
- **Tiempo Restante**: si la máquina está imprimiendo, en la tarjeta se mostrará el tiempo esperado restante de impresión.
- **Nombre**: muestra el nombre de la pieza que se está imprimiendo.
- **Barra de progreso**: se muestra una barra de progreso para identificar de una manera más visual el tiempo restante de impresión.
- **Extrusor**: se muestra la temperatura que tiene el extrusor actualmente y a su derecha la temperatura que hemos prefijado en el G-code o desde la propia máquina.
- **Cama**: si la impresora tiene ésta característica se mostrará otro apartado que indicará la temperatura actual de la cama y la temperatura que hemos prefijado con anterioridad.
- **Botonera**: por último mostraremos una botonera que cambia dependiendo del estado de la impresora. La botonera permite comenzar, pausar o cancelar una impresión.

En la figura 5.12 veremos un ejemplo de cómo mostramos los datos una impresora 3D sobre nuestra aplicación:

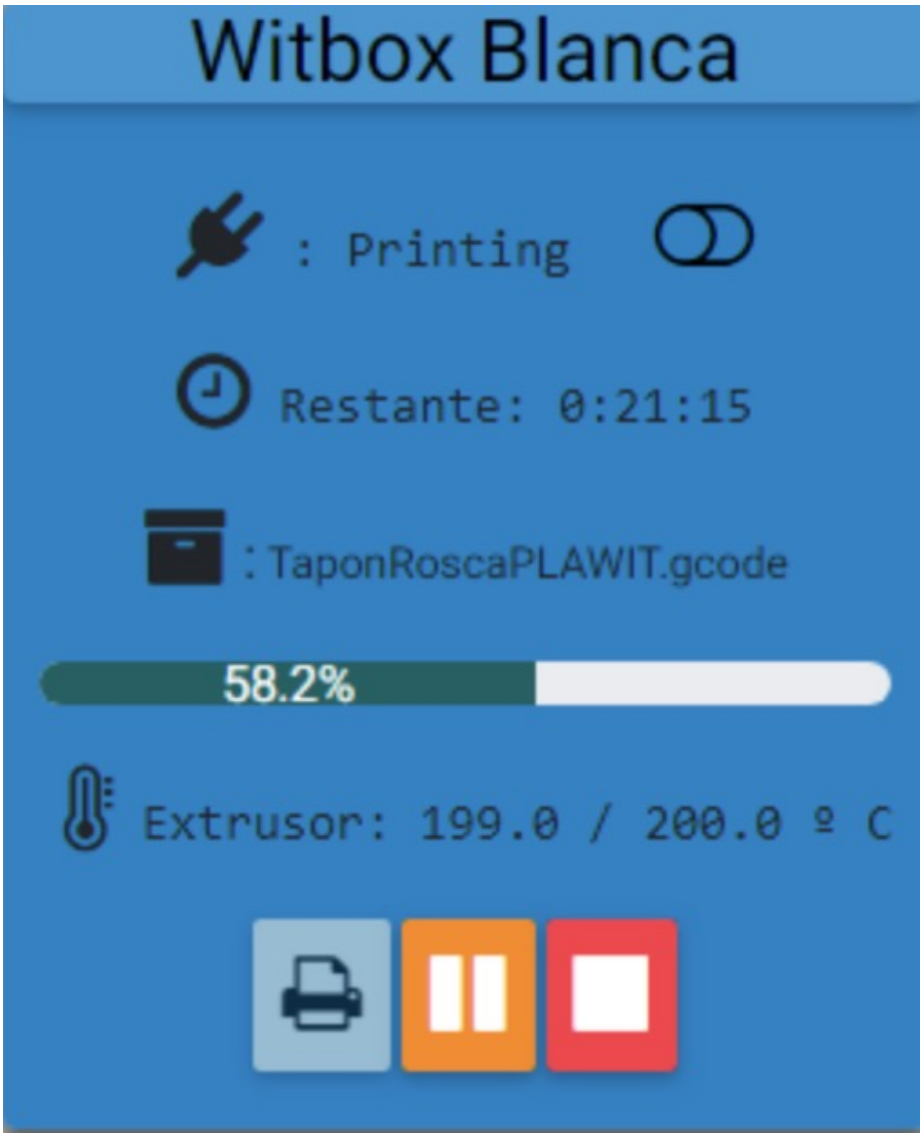

Figura 512: Ejemplo de de una máquina imprimiendo en la aplicación.

Con estas tarjetas alcanzamos uno de los objetivos de nuestra aplicación que es la de mostrar el estado de varias impresoras 3D desde una misma aplicación web. Estas tarjetas también nos permiten controlar el estado de cada máquina de una manera más rápida y sencilla. En la figura 5.13 podemos ver una vista general de la aplicación con siete máquinas funcionando simultáneamente.

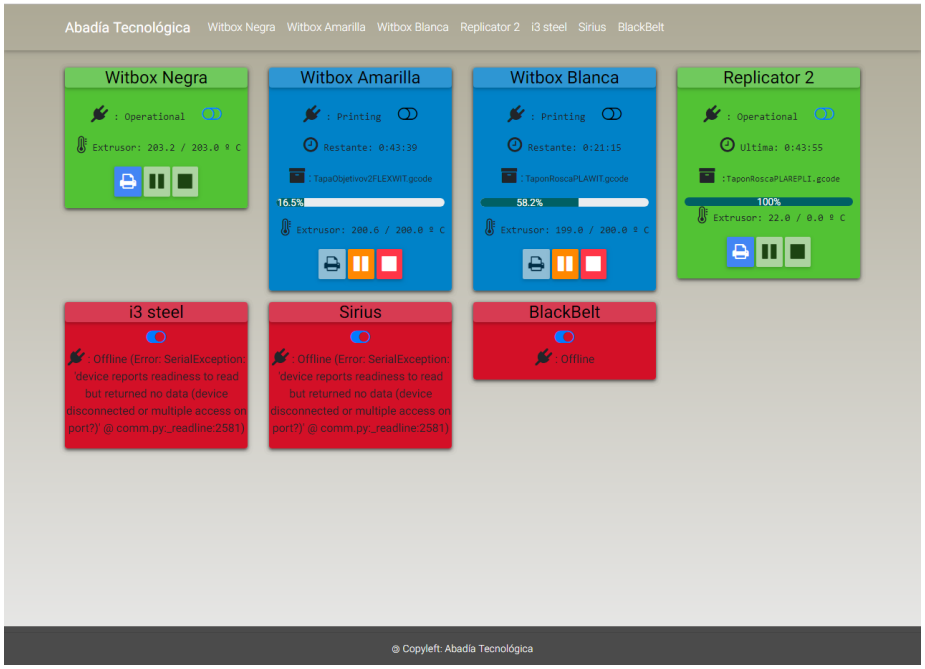

Figura 513: Vista completa de nuestra aplicación.

El color de cada tarjeta nos permite identificar los cuatro estados principales de cada impresora, que son:

- **Operativa**: la tarjeta tendrá un color verde para indicar que todo está correcto y está lista para imprimir. Podemos usar el botón de imprimir para comenzar la impresión siempre que tengamos un *G-code* cargado.

- **Imprimiendo**: la tarjeta tendrá una tonalidad azul y podemos usar los botones para pausar o cancelar la impresión en curso.

- **Pausa**: la tarjeta estará con una tonalidad anaranjada. En este estado podemos utilizar los botones para retomar la impresión o cancelarla.

- **Error**: en este caso la tarjeta estará de color rojo. El estado de error puede venir de dos motivos principales: el primero es que la impresora nos devuelva un error y en este caso nos dará información del error; y el segundo motivo es que puede que no esté lanzado el servicio de OctoPrint y en ese caso así nos lo hará saber la aplicación.

Identificación de usuarios

La aplicación desarrollada dispone además de un *Login* para poder llevar a cabo la identificación de los usuarios y poder así filtrar qué tipo de acciones pueden ejecutar cada perfil de usuario.

Para nuestro proyecto hemos identificado tres grupos de usuarios y hemos creado una pequeña base de datos para añadir los usuarios que necesitamos.

- **Admin**: es el administrador de la aplicación. Tiene acceso a todos los elementos de la aplicación y en un futuro será el encargado de añadir usuarios a la base de datos.
- **Operador**: tendrá acceso a todas las funcionalidades del sistema salvo a la creación de nuevos usuarios.
- **Visor**: este usuario será el que más restricciones tenga ya que no podrá utilizar la botonera de funcionalidades *(comenzar, pausar, cancelar)* y tampoco deberá tener acceso a la conexión y desconexión de las impresoras 3D.

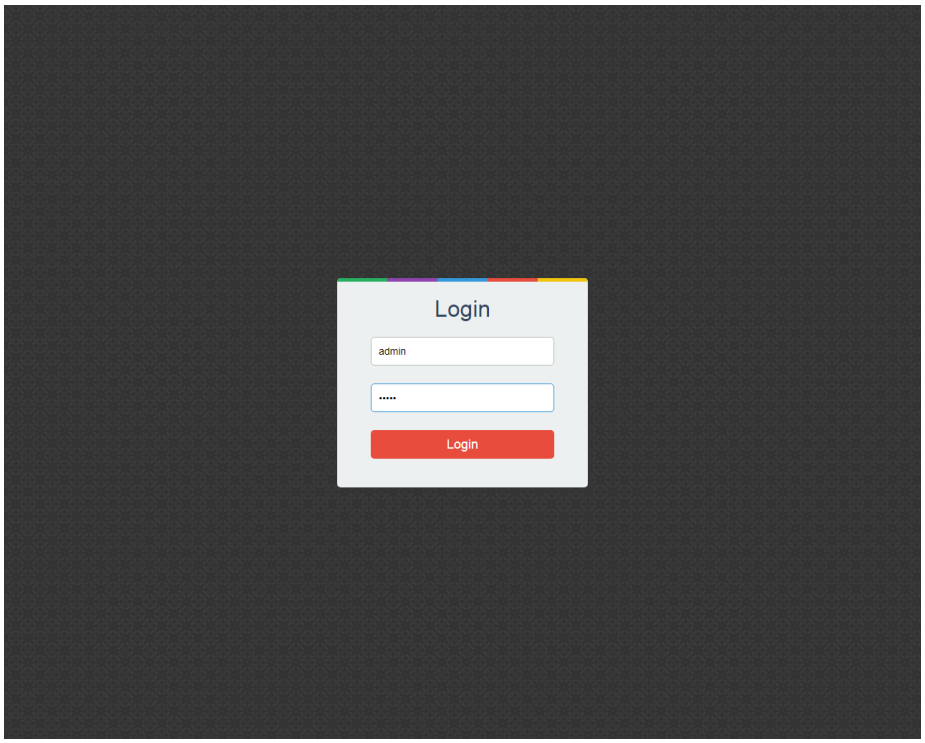

Figura 514: *Login* de nuestra aplicación web

Tal y como sabemos, dependiendo del tipo de usuario con el que se inicie sesión tenemos disponibles unas funcionalidades u otras. En la figura 5.15 se muestra un ejemplo de como se muestra la aplicación cuando hemos accedido como *Admin*.

Figura 515: Vista de la aplicación cuando eres *Admin*

A continuación vemos la vista de la aplicación cuando eres un *Visor*, en este caso no tenemos disponible ninguna funcionalidad.

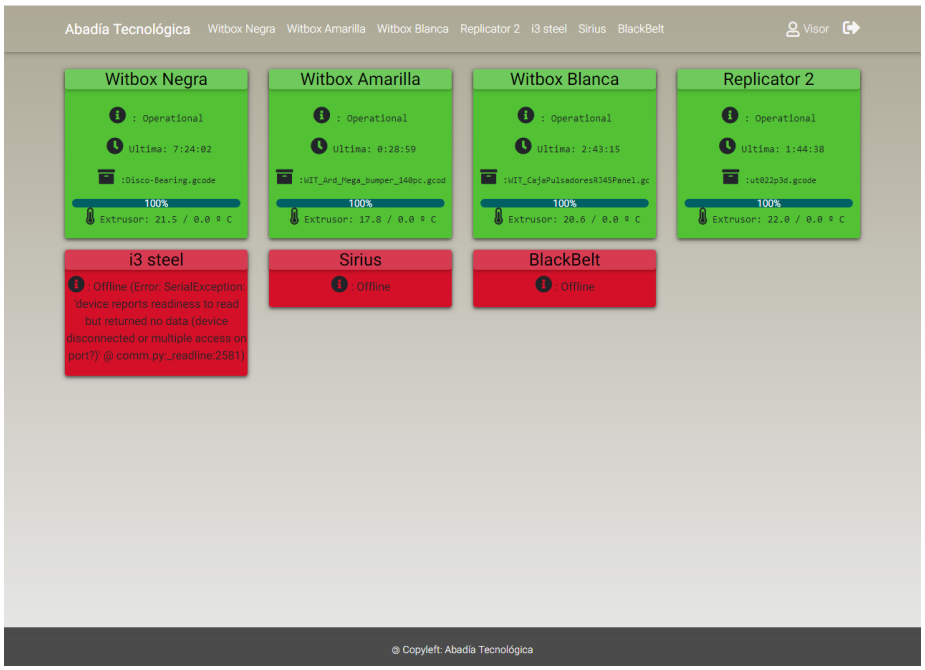

Figura 516: Vista de la aplicación cuando eres *Visor*

Trabajos relacionados

Existen multitud de trabajos y aplicaciones relacionadas con el mundo de la impresión 3D, pero casi todas ellas destinadas a gestionar una única impresora de manera similar a OctoPrint, como por ejemplo:

6.1. AstroPrint

AstroPrint [3] es una plataforma para controlar el uso de una impresora 3D desde la nube en cualquier tipo de dispositivo. Además crean su propio hardware desde el que poder controlar la impresora.

AstroPrint empezó su andadura como un *fork* de OctoPrint, pero ahora sigue su propio desarrollo.

Actualmente existen algunas alternativas a OctoPrint las cuales las podemos consultar en la siguiente página web [2].

Otro trabajo relacionado con el control de una impresora 3D y que podría ser un complemento del presente trabajo es el desarrollado en la Universidad de Burgos por **Omar Santos**.

6.2. 3DPrinterMonitoring

Este proyecto se encargaba de controlar cuando una impresión en curso era correcta respecto a un vídeo de referencia. De esta manera, mediante algoritmos de reconocimiento de imágenes, se podía detectar cuando la impresión era errónea y así poder cancelar la impresión con el fin de ahorrar tiempo y filamento.

Incluir una version de éste proyecto en nuestra aplicación web sería un gran complemento ya que el proyecto de Omar Santos se encargaría de detectar cuando la impresión no es correcta y con mi aplicación podemos cancelar la impresión en curso sin que el usuario tenga que hacerlo manualmente.

Conclusiones y Líneas de trabajo futuras

7.1. Conclusiones

Ahora que hemos concluido el desarrollo de nuestra aplicación web podemos decir que hemos conseguido cumplir todos los objetivos que nos propusimos al principio del desarrollo del proyecto.

He aprendido a usar herramientas como Flask, Nginx o SQLite y lenguajes como JavaScript y HTML con los que apenas tenía experiencia previa.

Por otra parte, también he adquiridos conocimientos sobre la impresión 3D ajenos a los ámbitos de la carrera.

También nos hemos encontrado alguna limitación: para el desarrollo de la totalidad de nuestro proyecto hemos tenido que solicitar la cesión de recursos a la empresa **Abadía Tecnológica**, la cual nos ha dejado las impresoras 3D y el hardware necesario para llevar a cabo este proyecto.

Además, una de las partes más complejas del desarrollo ha sido la de conseguir que funcione nuestra aplicación en el servidor; para ello hemos tenido que modificar varios *scripts* en Bash y no ha sido una tarea sencilla.

7.2. Líneas de trabajo futuras

Tal y como hemos comentado anteriormente, el desarrollo de nuestra aplicación no concluye aquí; la empresa que ha confiado en nosotros para el desarrollo de esta aplicación quiere continuar con el proyecto.

Cola de impresión

Tenemos previsto incluir una cola de impresión, que actualmente se está desarrollando por separado, en nuestro monitor. En el momento que entre un pedido nuevo compruebe que máquina es la mejor opción para imprimir ese pedido y le asigne automáticamente a una impresora en concreto. Como esta aplicación contará con una base de datos propia, sería interesante incluir nuestros usuarios en esa base de datos.

Creación de un Log

Otra cosa que sería muy interesante añadir es un *archivo log* que guarde los registros de nuestra aplicación, de manera que si alguna máquina produce algún error podamos entender rápidamente por qué se ha producido dicho error.

Diferenciación de usuarios

Actualmente, el usuario *Admin* y *Operador* tienen los mismos permisos, en un futuro próximo esto no será así.

La idea original es que el usuario *Admin* se encargue de añadir nuevos usuarios a la base de datos, mientras que el operador tenga acceso a todos los parámetros de las impresoras, pero no deberá tener acceso a la base de datos.

Bibliografía

[1] 3DPrint. Extrusor — flexion extruder: Take a lower-performing 3d printer to the industrial level. https://3dprint.com/143704/flexion-extruder-industrial/, 2019.

[2] Alternativeto. Alternatives to octoprint. https://alternativeto.net/software/octoprint/, 2019.

[3] AstroPrint. Productos de software que permiten a cualquiera crear, construir y aprender con impresoras 3d. https://www.astroprint.com/es, 2019.

[4] BBVAOPEN4U. Api rest— api rest: qué es y cuáles son sus ventajas en el desarrollo de proyectos. https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos, 2019.

[5] BIQU. Cama — new reprap 3d printer pcb heatbed mk2b 214*214*1.6mm heated thermistor bed hot plate for prusa and mendel thermistor bed. https://www.biqu.equipment/products/new-reprap-3d-printer-pcb-heatbed-mk2a-300-300-1-6mm-heated-bed-hot-plate-for-prusa-mendel, 2019.

[6] Norwegian Creations. G-code — an intro to g-code and how to generate it using inkscape. https://www.norwegiancreations.com/2015/08/an-intro-to-g-code-and-how-to-generate-it-using-inkscape/, 2019.

[7] Foosel. Github — octoprint. https://github.com/foosel/OctoPrint, 2019.

[8] Medium. Octopi/octoprint monitoring in node red. https://medium.com/@notsobadger/octopi-octoprint-monitoring-in-node-red-12dc7864b302, 2019.

[9] OctoPrint. Full remote control and monitoring. https://octoprint.org/assets/img/features/temperature-tab.png, 2019.

[10] OctoPrint. Octoprint — full remote control and monitoring. https://octoprint.org/, 2019.

[11] Simplify3d. Flexible filament — flexible. https://www.simplify3d.com/support/materials-guide/flexible/, 2019.

[12] Wikipedia. Bash — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Bash, 2019.

[13] Wikipedia. Bootstrap — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Bootstrap_(framework), 2019.

[14] Wikipedia. Flask — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Flask, 2019.

[15] Wikipedia. G-code — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/G-code, 2019.

[16] Wikipedia. Github — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/GitHub, 2019.

[17] Wikipedia. Html — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/HTML, 2019.

[18] Wikipedia. Javascript — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/JavaScript, 2019.

[19] Wikipedia. Jinja — wikipedia, la enciclopedia libre. https://en.wikipedia.org/wiki/Jinja_(template_engine), 2019.

[20] Wikipedia. Json — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/JSON, 2019.

[21] Wikipedia. Latex — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/LaTeX, 2019.

[22] Wikipedia. Nginx — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Nginx, 2019.

[23] Wikipedia. Putty — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/PuTTY, 2019.

[24] Wikipedia. Pycharm — wikipedia, la enciclopedia libre. https://en.wikipedia.org/wiki/PyCharm, 2019.

[25] Wikipedia. Python — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Python, 2019.

[26] Wikipedia. Sqlite — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/SQLite, 2019.

[27] Wikipedia. Sublime text — wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Sublime_Text, 2019.

[28] David Zotes. Monitor en tiempo real de un sistema de fabricación aditiva para-octoprint. https://github.com/AbadiaTecnologica/Monitor-en-tiempo-real-de-un-sistema-de-fabricacion-aditiva-para-Octoprint, 2019.