Universidad de La Laguna

Práctica 5

Servicio Web

Pierre-Simon Callist Yannick Tondreau

Introducción

En esta práctica se realiza la implementación de un servicio web que permita ejecutar las rutinas paralelas que se han desarrollado anteriormente. La implementación sigue el modelo Cliente/Servidor.

Backend

El backend ha sido desarrollado en NodeJS, usando el framework de Express. En ella se ha diseñado una API que permite la creación y ejecución de servicios en una máquina remota, y de la misma forma, obtener los resultados.

Todos los métodos de la API, devuelve un objeto en formato JSON o archivos singulares, por ejemplo la imagen resultado de los algoritmos desarrollados anteriormente.

A continuación se mostrarán los métodos:

Crear Servicio

Para crear un servicio es necesario realizar cuatro llamadas diferentes a la API.

1. La primera crea el servicio en la base de datos.

```
/**
 * PUT - Crea el algoritmo en la base de datos
 */
router.put('/put/service/:service', async (req, res, next) => {…
})
```

2. Las siguientes añaden los archivos *Makefile* y el archivo fuente.

```
/**
 * PUT - Crea el archivo fuente del algoritmo :service en la base de datos
 */
router.put('/put/service/:service/source/:filename', async (req, res, next) => {…
})

/**
 * PUT - Crea el Makefile del algoritmo :service en la base de datos
 */
router.put('/put/service/:service/makefile', async (req, res, next) => {…
})
```

3. La última compila el servicio en el entorno remoto.

```
/**
 * POST - Compila un servicio que tiene Makefile y archivo fuente
 */
router.post('/post/compile/:service', async (req, res, next) => {…
})
```

Ejecutar Servicio

Para ejecutar un servicio se realiza una llamada a la API, en la cual se deberá de rellenar cada uno de los parámetros especificado en el servicio. Y devuelve el resultado en formato JSON.

```
/**
 * POST - Ejecuta un servicio y devuelve el resultado
 */
router.post('/post/:service', async (req, res, next) => {…
});
```

El resultado es el siguiente:

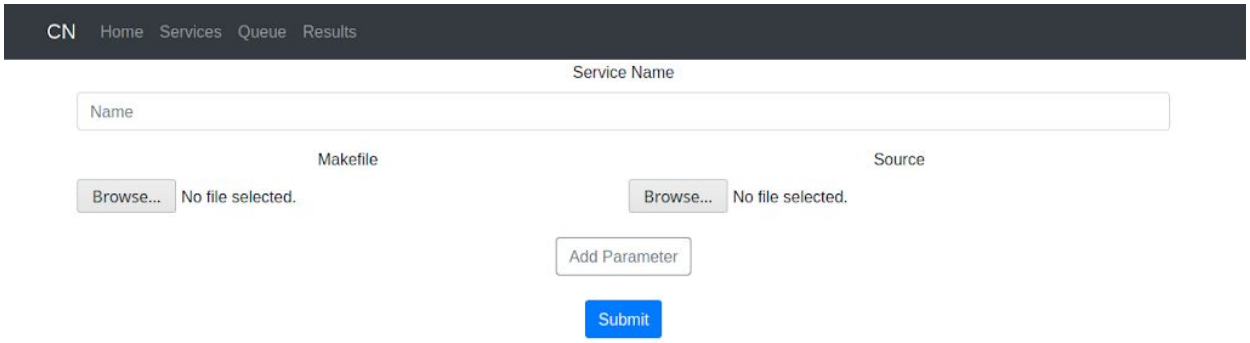
```
res.status(200).json({
    date: Date(timestamp),
    vars: exec_params,
    result: {
        console: console_result,
        file: result_file
    }
})
```

Los métodos expuestos anteriormente son los principales, también se dispone de métodos para obtener los servicios almacenados, así como para obtener los resultados de una ejecución en concreto.

Frontend

El frontend ha sido desarrollado usando el framework React. en ella se podrá crear y ejecutar servicios mediante el uso de la API que se ha desarrollado.

Crear Servicio



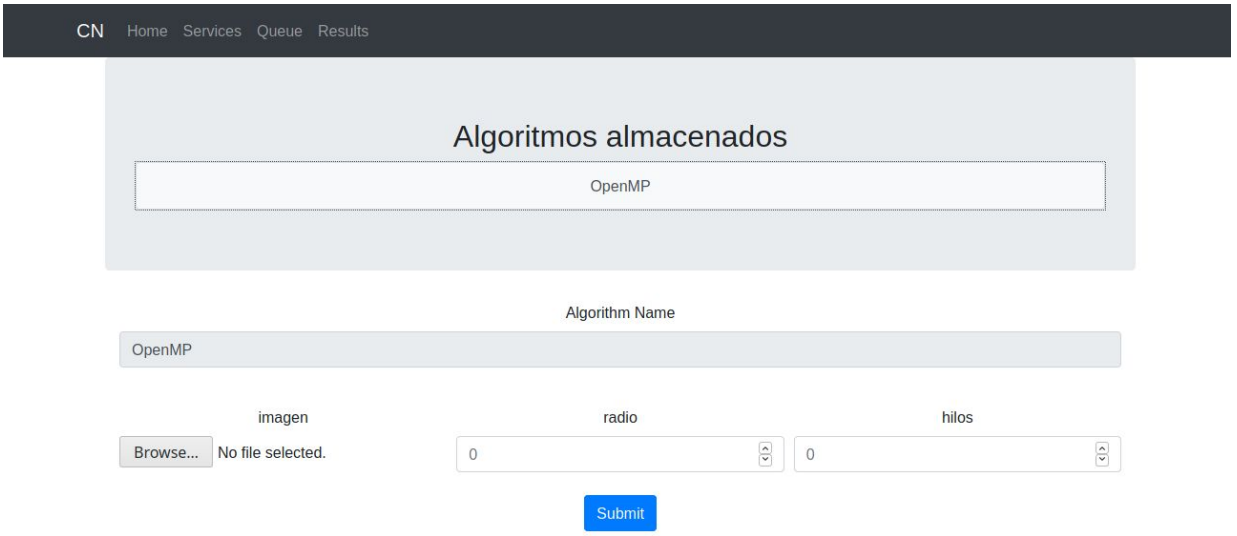
Para crear un servicio se debe de disponer de su archivo ***Makefile*** en donde se especifican dos métodos *build* y *run*. El método build se usará para compilar el programa (en caso de no necesitar compilación, se deja vacío). El método run se usará para ejecutar el programa (se deben de pasar los parámetros como variables al comando make). Si el programa devuelve un archivo se especifica adicionalmente el parámetro result.

```
build:
	g++ src/openmp.cpp -o openmp.run `pkg-config --libs opencv` -fopenmp -std=c++11

run:
	bin/openmp.run $(image) $(radio) $(hilos) $(result)
```

Ejecutar Servicio

Para ejecutar un servicio, seleccionamos el deseado y se mostrará los parámetros a rellenar para la ejecución del mismo.



Con los parámetros rellenados, podemos proceder a la ejecución, de forma seguida se mostrará el resultado en una ventana modal.