



PROYECTO INTEGRADO

Kubectl4everyone



16 DE JUNIO DE 2022
MARK BLAZQUEZ

Contenido

1.	Estudio del problema y análisis del sistema	2
2.	Introducción.....	3
3.	Funciones y rendimientos deseados.....	4
4.	Objetivos	5
5.	Planteamiento y evaluación de diversas soluciones.....	6
6.	Justificación de la solución elegida.....	7
7.	Modelado de la solución.....	8
a.	Recursos humanos	8
b.	Recursos hardware	8
c.	Recursos software.....	8
8.	Planificación temporal. Etapas.....	9
9.	Ejecución del proyecto y elaboración de la documentación técnica.....	10
10.	Fase de pruebas con la creación de una batería de pruebas (tanto para descubrir errores lógicos como posibles mejoras en la ejecución) que corroboren de forma razonable el buen funcionamiento del sistema.	10
11.	Documentación del sistema.....	11
a.	Introducción a la aplicación	11
b.	Manual de Instalación.....	11
c.	Manual de usuario.....	12
d.	Manual de administración	18
12.	Conclusiones finales.....	19
13.	Grado de cumplimiento de los objetivos fijados.	20
14.	Bibliografía	21

1. Estudio del problema y análisis del sistema

Debido a mi temprana incursión en el mundo laboral en el negocio de venta de alimentación a clientes y mi constante cercanía al mundo tecnológico siempre he encontrado tedioso e improductivo tener que tomar nota de los encargos en una libreta, como si estuviésemos en siglo xx, con sus problemas como que no entiendan la letra, se rompa, ensucie y quede ilegible etc...

Algunos de los dueños de estos negocios se han atrevido a intentar poner solución de forma creativa con un excel pero el resultado es terrible.

Es aquí donde mi idea aparece, y va a intentar solucionar esto con una sencilla página web que permita tomar notas de forma fácil y ordenada, controlando los fallos y accesible desde cualquier tipo de dispositivo.

Pero es evidente que todo esto deberá de estar hospedado en un servidor para que el cliente pueda acceder, el cual deberá de ser seguro a nivel de ofrecer las páginas y para ello aparecerá otra aplicación para brindar herramientas de control sobre el servidor al desarrollador

2. Introducción

El proyecto estará compuesto de dos páginas en react que ofrecen solución a página de información y gestión de pedidos de un asador, estas estarán en kubernetes, y su backend con node que será el api rest.

Para facilitar la vida al desarrollador se hará una aplicación en symphony que gestionará el estado de kubernetes, y para ello se debe de hacer del servidor un lamp y hospedarla en este.

Todas las páginas serán seguras y se accederán a ellas por nombre.

El servidor tendrá un firewall para protegerlo de acciones indeseadas por parte de usuarios externos.

Por último, la aplicación de react será portada para la ejecución nativa en los sistemas operativos móviles

3. Funciones y rendimientos deseados

Este proyecto debe de satisfacer dos hechos primordiales

a. Aplicación de gestión de pedidos

En esta aplicación se debe de poder de insertar, borrar información de los pedidos, y para tener controlado que si en caso de existir dos páginas abiertas a la misma vez no existan disonancias entre una y otra por la inserción de pedidos y el no recargar la se aplicara un auto recarga a la sección de los pedidos.

Se intentará portear de forma nativa esta aplicación a sistemas móviles.

b. Aplicación de gestión de kubernetes y otros servicios

Con esta aplicación se va a intentar lograr obtener información de que solamente sería accesible mediante terminal, al igual que se le va a añadir funciones de estas como controlar el estado de servicios y funciones vinculadas

4. Objetivos

Los servicios que ofrecerá una vez acabado son:

- Una aplicación de gestión de pedidos, que ofrezca la información actualizada sin necesidad de recargar la página el usuario
- Portear esta aplicación a sistema operativo móvil
- Elaboración de aplicación que le brinde una herramienta de control al desarrollador de servicios y kubernetes para evitar el uso de terminal
- Paginas accesibles por nombre
- Paginas seguras
- Servidor seguro con un firewall

5. Planteamiento y evaluación de diversas soluciones

En el basto mundo del desarrollo y gestión de sistemas hay tantas posibilidades de desarrollar una idea como tiempo tengas para intentarlo, pero yo tenía claro el uso de algunas.

Y aunque se tenga claro el uso de las tecnologías se pueden utilizar varias formas para llevarlo a cabo, y algunas de las decisiones a tomar son:

- Node port vs ingres
A la hora de implementar el servicio que nos permitirá la entrada a las aplicaciones gestionadas por kubernetes se nos presenta la opción de crearlo de varias formas load balancer y utilizar un ingres o node port y directamente hacer un port forward
- Backend node vs consultas Ajax
Para hacer consultas a un documento y obtener, borrar e insertar información voy a utilizar javascript pero se me presenta la opción de utilizar un servidor node y gestionarlo todo desde ahí o con jquery y Ajax hacer peticiones
- Login con procedure en doctrin en symfony o sin framework en php
A la hora de implementar el inicio de sesión se valoraba el uso de procesos en vez de simples consultas, pero para hacer el inicio podía hacerlo de manera externa con php y aplicar este proceso o symfony y tener que modificar como funciona doctrine
- Ufw vs iptables
Para la gestión de firewall se presentan dos opciones, pero mi indecisión viene dada en utilizar algo de más de bajo nivel como iptables o ufw que básicamente es un framework de iptables

6. Justificación de la solución elegida.

- Node port vs ingres

Node port será la elección, consultando la documentación oficial de kubernetes a parte de otras páginas de usuarios avanzados de esta herramienta hacen hincapié en lo buena que es la solución de un ingress, pero si el proyecto es de proporciones épicas y se van a hospedar muchas y diferentes páginas, como no es mi caso la solución de node port es eficaz y más que suficiente

- Backend node vs consultas Ajax

Node es el elegido por la razón de que proporciona herramientas como fetch que deja en pañales a Ajax y el creciente uso de api rest me provoca la suficiente curiosidad como para utilizarlo en mi proyecto

- Login con procedure en doctrin en symfony o sin framework en php

La solución a utilizar será php vanilla , esto es porque al consultar paginas expertos dicen que usar procesos con doctrine no tiene sentido

- Ufw vs iptables

Como puntuizaba en la anterior sección ufw es básicamente un framework de iptables y es por ello que lo voy a utilizar , para aumentar la rapidez y disminuir la tediosa forma con la que se trabaja con iptables

7. Modelado de la solución

a. Recursos humanos

A diferencia de otros institutos que el proyecto se realiza entre un grupo de tres personas, este se realizará única y exclusivamente por mi persona.

b. Recursos hardware

Para el uso de kubernetes en forma de cluster se necesita de dos pcs, es evidente que se necesita de periféricos como ratón, teclado y monitores además de un buen espacio bien ventilado e iluminado y por supuesto una buena silla ergonómica que mantenga mis lumbares en una buena forma.

c. Recursos software

A nivel de software se necesitarán todas las herramientas con las que se va a trabajar:

- Procesador de textos: en mi caso elegiré un ide que permite extensiones y facilita el trabajo
- Node: para el back end
- Npm: la herramienta que permite la descarga y uso de módulos varios en node y la puesta en marcha de proyecto de react
- Linux(ubuntu), apache, mysql y php además de symfony: para nuestra app de gestión.
- Ufw: para firewall.
- Openssl, lets encrypt, no ip: para dns y certificados.
- Docker, kubernetes: herramienta para hospedar la página web de gestión de comercio
- Expo, yarn: desarrollo móvil
- Nginx: servidor web para los deployment

8. Planificación temporal. Etapas.

- Marzo
 - Puesta en marcha de los equipos
 - Creación de cluster de kubernetes
- Abril
 - Desarrollo apps react y backend node
- Mayo
 - Terminar app react node e implementación en kubernetes (hasta el 18)
 - App Symphony
 - Login con procesos
- Junio
 - Dns, certificados ssl, test de fallos a través de dns
 - Port react native

9. Ejecución del proyecto y elaboración de la documentación técnica
10. Fase de pruebas con la creación de una batería de pruebas (tanto para descubrir errores lógicos como posibles mejoras en la ejecución) que corroboren de forma razonable el buen funcionamiento del sistema.

Ambas partes serán anexionadas al final de la documentación donde recojo todos y cada uno de los aspectos tratados de la forma más extensa posible y contemplando todos los errores que iban causando, mostrando su solución

Ver anexo pág. 22

11. Documentación del sistema

a. Introducción a la aplicación

Como ya he comentado la aplicación de gestión de comercio radica en un index que proporciona información al usuario, y un botón que nos lleva a la página para gestionar los pedidos

Una vez ya en la propia página de gestión nos encontraremos con un formulario de inserción de pedidos, que una vez que esta añadido nos muestra el pedido y aparece un botón para poder borrarlo.

De la misma forma esta app estará replicada, pero de forma nativa en la aplicación móvil, que dispone de una pantalla que muestra los pedidos, otra con un formulario de inserción y otra con un formulario de borrado.

Por otro lado, tenemos la aplicación para controlar los servicios, que para llegar aquí debemos de hacer un login.

Una vez en la main nos encontramos con un menú de navegación que nos lleva a las distintas rutas donde controlaremos el estado de los pod , service, deployment, que aquí nos permitirá aumentar los números de pod , replicaset y de nginx que permite parar y subir el servicio

b. Manual de Instalación

Debido a que las aplicaciones son páginas web no necesitan de instalación a excepción de la de Android que se distribuiría en apk que solo es permitir el acceso a aplicaciones de orígenes desconocidos e instalar, aunque de momento como está en fase beta se debe de descargar expo y utilizarla desde ahí

c. Manual de usuario.

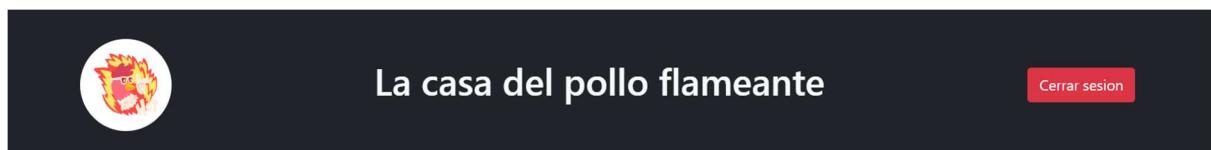
Aplicación asador

La primera página es el index que muestra información sobre el comercio y otros datos, que su principal uso es mantener al usuario informado

Pero aquí lo importante es el botón de inicio de sesión que lo que permite es ir a la página de gestión de pedidos, que conducía a un inicio de sesión de a través de google, pero por problemas con la conectividad de kubernetes con internet ha sido deshabilitado



Una vez ya redirección a la página veremos que en la cabecera se encuentra el botón de cerrar sesión



Podemos visualizar también nel formulario de inserción de datos que tiene dos campos numéricos el de patatas y pollo y uno de texto que es el de nombre

Pollo
Patatas
Nombre

[Registrar](#)

En caso de no haber pedidos se mostrará en pantalla

Pedidos

no hay Pedidos

Una vez que el pedido se inserta se muestra por pantalla, y aparece un botón de borrado que nos permite borrar este pedido

Pedidos			
Pollo: 2	Patatas: 2	Nombre: mark	Borrar

Lo bueno de esta aplicación es que nos permite tener constantemente actualizadas las páginas, aunque tengamos diferentes páginas en distintos dispositivos insertando y borrando, lo renderiza y consulta la información cada segundo por lo que no hace falta recargar la página

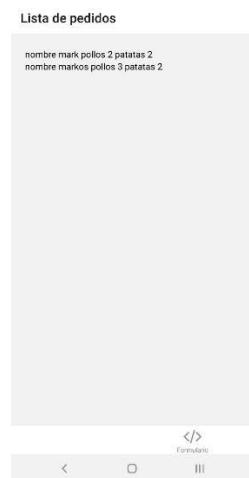
Aplicación de móvil

Para el desplazamiento de las diferentes pantallas debemos de tener en cuenta el menú de abajo en la pantalla



Que no aparecerá el ícono de donde estemos para evitar confusión

La primera pantalla será la que muestre los pedidos



La segunda pantalla será la de añadir que debemos de tener en cuenta el meter un id alfanumérico, a diferencia de la aplicación web

Añadir

nombre	
pollo	
potatas	
id	

ENVIAR

</> </>
Lista de pedidos Añadir
< < <<

Por ultimo tenemos la pantalla de borrado que se debe de introducir el id del pedido, que lo mostrara la lista de estos

Borrar

id	
----	--

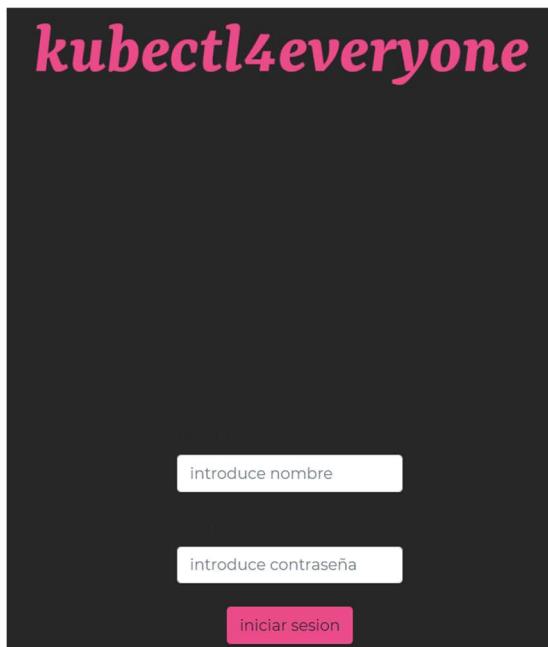
ENVIAR

</> </>
Lista de pedidos Añadir
< < <<

Aplicación symfony

Como es una aplicación para el desarrollador está ubicada en un puerto concreto y por https, el 8888, para que solo acuda a esta pag quien sepa de su existencia.

Lo primero que encontraremos será un login



Este login nos conduce la página principal, pero si nos equivocamos se re direcciona a la página de error.



El botón nos permitirá intentar de nuevo el inicio de sesión.

Ya en el menú podemos ver que hay un botón para cerrar sesión, porque si no permanecerá abierta

The screenshot shows a dark-themed web application interface. At the top right, there is a pink rectangular button labeled "Cerrar session". Below the header, there is a horizontal navigation bar with five items: "POD", "SERVICE", "DEPLOYMENT", "REPLICASET", and "NGINX".

En cada uno de los apartados se nos ofrecerá información de acerca de esa sección y podemos ver un botón para volver al inicio

Pod			
READY	STATUS	RESTARTS	
1/1	Running	1_(13h ago)	
1/1	Running	1_(13h ago)	
m9d	1/1	Running	1_(13h ago)
snl	1/1	Running	1_(13h ago)
5	1/1	Running	1_(13h ago)

Volver a inicio

En el apartado de deployment encontramos un formulario que nos permite cambiar el número de pod de un deployment.

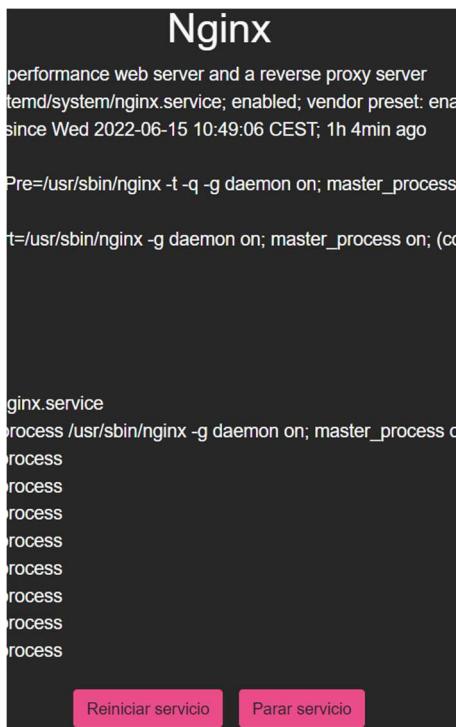
The screenshot shows a dark-themed form titled "Formulario de cambio de numero de pod". It contains two input fields: "nginx-pagina con index" and "introduce numero de pod". Below the inputs is a pink rectangular button labeled "modificar".

Para no preocuparnos de poner bien el nombre de ese deployment será un seleccionable



Y ya solamente debemos de clicar en el botón de modificar.

Otra sección interesante es la de nginx que permite parar y recargar el servicio a través de los botones con tan solo clicar al botón deseado



d. Manual de administración

Para la administración de estas aplicaciones se puede observar el anexo donde se aborda la información de forma detallada, pero debemos de tener en cuenta que la aplicación de react y node van en kubernetes por lo que siempre debemos de tener bien configurado el yaml para no tener una imagen de docker que pertenezca a la última versión.

En mi caso al estar todo configurado ya solo debemos de tener conectado ambos servidores por ethernet a la red y encendidos y con eso ya las aplicaciones están listas para consumirse

Para la aplicación en react native (móvil) debemos de tener expo cli en nuestro pc lo que permitirá realizar los nuevos ajustes necesarios y realizar el compilado para distribuirlo en las diferentes tiendas o como yo de momento lo tengo hacer de expo el emulador y que el móvil lo pueda representar.

Para la aplicación de apache debemos de tener presente que al ir seguro debemos de iniciar el servicio cada vez ya que nos pedirá la clave del certificado ssl

12. Conclusiones finales.

Como conclusión me gustaría destacar que el uso de herramientas como kubernetes, docker tienen una curva de aprendizaje muy pronunciada y como se puede ver en el proyecto de forma rápida se pueden poner en producción de forma ‘rápida’ y sin apenas ‘conocimiento’ proyectos del calibre que uno guste.

Pese al avance continuo de tecnologías e ingente cantidad de lenguajes abrumadores, las posibilidades de realizar un proyecto a partir de una idea pueden realizarse de forma fácil ya sea con uno u otro.

Nos encontramos en una era donde aprender una tecnología nueva es más fácil que nunca, solo depende de nuestra gana y buen hacer.

Se debe ser adaptable y conocer, aunque sea, los lenguajes más importantes (en mi caso, que la programación es lo que me interesa, podríamos extrapolar en el caso de la administración de sistemas) y lo que son capaces de ofrecernos, atascarse en un lenguaje denota que la pasión por este mundillo y las capacidades intelectuales brillan por su ausencia.

Pese a que la especialización es la mayor herramienta para la progresión laboral no ser capaces de valorar y conocer lo que otras tecnologías ofrecen provoca el fenómeno conocido como ‘cuñado’ que es hablar sin tener ni idea, que cada vez está más extendido, por lo que estaría fenomenal que lo tuviésemos en cuenta e intentar no caer en él.

13. Grado de cumplimiento de los objetivos fijados.

El proyecto ha sido finalizado completamente, todas las ideas concebidas desde el anteproyecto han sido implementadas, incluso algunas funciones han sido añadidas y mejoradas utilizando tecnologías más potentes.

Aunque ha surgido muchos problemas desde su concepción y aplicación, la gran mayoría han podido ser solventados, a excepción de algo que por tiempo no han sido corregidos, como el problema de conexión de los pod con internet o un mayor pulido de interfaces gráficas.

Propuesta de modificaciones o ampliaciones futuras del sistema implementado.

Como mejoras futuras para el proyecto y lo más importante como mejora a mi conocimiento sobre nuevas tecnologías me gustaría implementar el uso de la base de datos con la aplicación de symfony y así conocer el uso de doctrine lo que gestiona, que por requerimiento del profesorado no ha podido ser utilizado en el inicio de sesión.

Otra función que me hubiera gustado implementar es la elaboración de los yaml de creación de deployment desde la web, ya sea subir un archivo o desde la misma web crear uno que se guarde en servidor y poder ponerlo en producción de manera fácil

En el port de la aplicación de react a react native los formularios no están validados por lo que la inserción de datos vacíos se permite, ver la solución a este fallo y corregirlo ya que no es tan fácil como html

14. Bibliografía

- <https://reactnative.dev/>
- <https://symfony.com/doc/current/index.html>
- <https://kubernetes.io/>
- <https://reactjs.org/>
- <https://nodejs.org/en/docs/>
- <https://www.noip.com/>
- <https://www.docker.com/blog/how-to-use-the-official-nginx-docker-image/>
- https://hub.docker.com/_/httpd
- <https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu-18-04>
- <https://serverfault.com/questions/828130/how-to-run-nginx-ssl-on-non-standard-port>
- <https://letsencrypt.org/>

ANEXO

Tabla de contenido

1.	Elección de hardware e instalación del sistema operativo.....	25
2.	Configuración de la red.....	26
3.	Instalación de kubernetes en forma de cluster	27
4.	Desarrollo de la aplicación de encargos con react y node.....	30
4.1.	Servidor back-end	30
4.1.1.	Ruta que obtiene los pedidos	31
4.1.2.	Ruta que guarda los pedidos (de un formulario a este archivo mencionado).....	31
4.1.3.	Ruta de borrado de pedidos	31
4.1.4.	Index.....	32
4.1.5.	Ruta loguin Google.....	32
4.1.6.	Ruta redirección a página pedidos.....	33
4.1.7.	Ruta de cerrar sesión	33
4.2.	Front end.....	34
4.2.1.	Index.....	34
4.2.2.	Página pedidos	38
5.	Creación del contenedor docker.....	44
5.1.	Servidor web	44
5.2.	Back end.....	47
6.	Implantación de la aplicación en kubernetes	49
6.1.	Front end.....	49
6.2.	Back end.....	52
7.	Instalación y configuración del entorno de desarrollo symphony (para la página de gestión de kubernetes y servicios)	53
7.1.	Instalación de php.....	53
7.2.	Instalación de servidor web	54
7.3.	Instalación mysql	55
7.4.	Instalación del framework	57
8.	Desarrollo de la página de login para la página de gestión de servicios y kubernetes	58
9.	Desarrollo de la página de gestión de servicios, kubernetes con symphony	61
9.1.	Creación de paginas	61
9.2.	Página de nginx	68
9.3.	Paso de aplicación a producción en apache	71

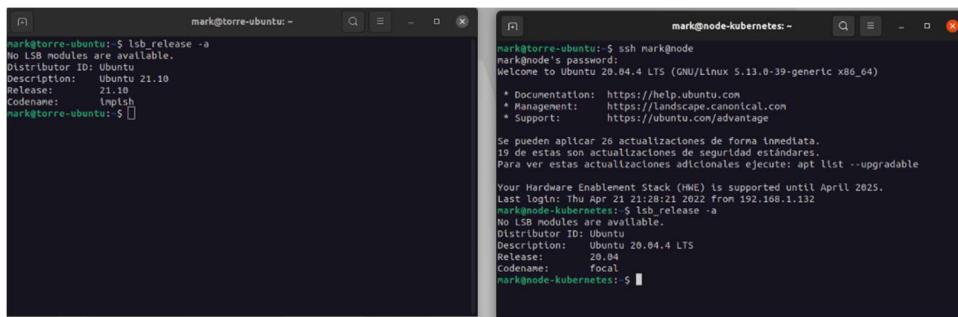
9.4.	Cookie para mantener sesiones.....	76
9.5.	Cierre de sesión.....	77
9.6.	Funcionalidad extra: cambio de numero de pod.....	78
10.	Dns público: no ip	81
10.1.	Adopción del servicio	81
10.2.	Uso en la app de apache	83
10.3.	Uso en las apps de kubernetes	84
11.	Instalación de certificados en apache, nginx y node	85
11.1.	Nginx(docker).....	85
11.2.	Apache	89
11.2.1.	Docker	89
11.2.2.	Servidor	91
11.3.	Node.....	94
12.	Instalación y configuración de firewall	98
13.	Port de la aplicación a dispositivos móviles.....	100
13.1.	Explicación del entorno.....	100
13.2.	Mostrar datos de api.....	105
13.3.	Añadir un screen y su posición en el menú	106
13.4.	Añadir.....	107
13.5.	Eliminar	109
13.6.	Añadiendo recarga de pagina	111
13.7.	Redirección de página cuando se hace el formulario	111
13.8.	Problemas con https(certificados node).....	112

1. Elección de hardware e instalación del sistema operativo

Para el desarrollo de este proyecto he elegido hardware ya en mi disposición en este caso de un i7 de 6 gen desbloqueado ,24 de ram, una gráfica nvidia dedicada al entorno laboral (gama quadro) que dispone de tres salidas display port elegido así por su función de cuando el monitor se apaga la tarjeta lo detecta y esta le pasa la señal al sistema operativo permitiendo una gestión de pantallas más eficiente ya que estas se desactivan a diferencia de otras conexiones de video ,en mi caso utilzo 3 monitores así que viene fenomenal, un ssd de 128 pequeño pero suficiente , y por ultimo pero no menos importante unas bonitas luces que le aportan un rendimiento extra ;)

En el node (equipo cedido por el instituto, por la necesidad de hacer un cluster en kubernetes a deseó expreso de un miembro del profesorado y nula disposición a utilizar mi ordenador principal para ello) nos encontramos con un i5 de 7 gen, 8 de ram y 128 ssd y grafica nvidia

El sistema operativo elegido va a ser Ubuntu por su compatibilidad con los drivers de las tarjetas gráficas que utilizo, que son nvidia, y por su gestión de ventanas más cómoda visualmente hablando. Nos aseguraremos del funcionamiento de ssh comprobamos su instalación con el comando status de systemctl y si no instalamos (aunque sea de perogrullo) ya que es como gestionaremos el node desde el master



The image shows two terminal windows side-by-side. The left window is titled 'mark@torre-ubuntu:' and the right window is titled 'mark@node-kubernetes:'. Both windows display the output of the command 'lsb_release -a'. The output includes details about the distribution (Ubuntu), release (21.10 or 20.04.4 LTS), and codename (Impish or focal). The right window also shows a welcome message for Ubuntu 20.04.4 LTS and information about the Hardware Enablement Stack (HWE) support until April 2025.

```
mark@torre-ubuntu:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 21.10
Release:        21.10
Codename:       impish
mark@torre-ubuntu:~$ 

mark@node-kubernetes:~$ ssh mark@node
mark@node's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Se pueden aplicar 26 actualizaciones de forma inmediata,
19 de estas son actualizaciones de seguridad estándares.
Para ver estas actualizaciones adicionales ejecute: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Aug 11 20:28:21 2022 from 192.168.1.132
mark@node-kubernetes:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.4 LTS
Release:        20.04
Codename:       focal
mark@node-kubernetes:~$
```

2. Configuración de la red

La configuración de red en Ubuntu a diferencia de Debian se realiza a través de netplan de forma predeterminada así que para modificarla deberemos de cambiar los ajustes en el directorio netplan ubicado en etc y seleccionaremos el adaptador disponible



Como vemos solo he realizado la configuración en ipv4 y esto es por el hecho de que mi isp no me proporciona una dirección ipv6

Recordemos que estos equipos van a ser servidores por lo que necesitamos que su ip sea estática

Sudo netplan apply para aplicar los cambios y vemos el resultado

```
harold@torre-ubuntu:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
    2: enp0s31f6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
        group default qlen 1000
        link/ether 4c:cc:0a:ff:fe:03 brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.127/24 brd 192.168.1.255 scope global noprefixroute enp0s31f6
            valid_lft forever preferred_lft forever
            linklayer brd ff:ff:ff:ff:ff:ff
            link/ether 18:31:0f:dd:b1:4e brd ff:ff:ff:ff:ff:ff
            inet 192.168.1.128/24 brd 192.168.1.255 scope global noprefixroute enp4s0
                valid_lft forever preferred_lft forever
                linklayer brd ff:ff:ff:ff:ff:ff
                link/ether fe:80::1a:11:bf:ff:fed:bie/64 scope link
                    valid_lft forever preferred_lft forever

mark@node-kubernetes:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
            linklayer brd ff:ff:ff:ff:ff:ff
            link/ether 18:31:0f:dd:b1:4e brd ff:ff:ff:ff:ff:ff
            inet 192.168.1.128/24 brd 192.168.1.255 scope global noprefixroute enp4s0
                valid_lft forever preferred_lft forever
                linklayer brd ff:ff:ff:ff:ff:ff
                link/ether fe:80::1a:11:bf:ff:fed:bie/64 scope link
                    valid_lft forever preferred_lft forever
```

3. Instalación de kubernetes en forma de cluster

Esta documentación la he elaborado después del proceso de instalación por lo que las capturas de pantalla se han extraído del historial de comandos de bash que se haya en bash_history por eso no aparecen los numeritos que en comando history si, lo comento por la disonancia que puede verse al no presentarse el nombre del equipo

```
mark@torre-ubuntu:~$ cat $HOME/.bash_history
```

Para instalar kubernetes en forma de clúster debemos de comenzar instalando kubelet, kubeadm and kubectl y para ello debemos de agregar su repositorio

```
sudo apt -y install curl apt-transport-https
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list
sudo apt update
sudo apt -y install vim git curl wget kubelet kubeadm kubectl
```

Revisaremos la instalación consultando la versión

```
mark@torre-ubuntu:~$ kubectl version --client && kubeadm version
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.5", GitCommit:"c285e781331a378597f436942c05c5641ce8a9e9", GitTreeState:"clean", BuildDate:"2022-03-10T15:58:47Z", GoVersion:"go1.17.8", Compiler:"gc", Platform:"linux/amd64"}
kubeadm version: VersionInfo{Major:"1", Minor:"23", GitVersion:"v1.23.5", GitCommit:"c285e781331a378597f436942c05c5641ce8a9e9", GitTreeState:"clean", BuildDate:"2022-03-10T15:57:37Z", GoVersion:"go1.17.8", Compiler:"gc", Platform:"linux/amd64"}
```

Los siguientes pasos consistirán en deshabilitar el swap que nos viene bien porque mis equipos se van a reiniciar, y habilitar y añadir algunas reglas para el kernel y sysctl

```
sudo sed -i '/ swap / s/^(\.*\$)//#\1/g' /etc/fstab
sudo swapoff -a
sudo modprobe overlay
sudo modprobe br_netfilter
sudo tee /etc/sysctl.d/kubernetes.conf<<EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip_forward = 1

EOF

sudo sysctl --system
```

Lo siguiente será habilitar el correr los contenedores en docker

```
sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update
sudo apt install -y containerd.io docker-ce docker-ce-cli
sudo mkdir -p /etc/systemd/system/docker.service.d
sudo tee /etc/docker/daemon.json <<EOF

{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF

sudo systemctl daemon-reload
sudo systemctl restart docker
sudo systemctl enable docker
```

Es momento de habilitar kubelet y descargar nuestro control plane y algún detalle más y por fin hacer el comando que nos permitirá vincular nodos a nuestro cluster

```
sudo kubeadm init
```

Esto nos volcará un comando que nos volcará un token que servirá de vinculación y que deberá de introducirse en el node

```
166 kubeadm join 192.168.1.132:6443 --token rkq458.pvix237wpy2f824z --discovery-token-ca-cert-hash sha256:9
```

Podemos ver que apunta a la ip del master, por último, debemos de copiar y pegar un par de órdenes que nos vuelca para declarar el directorio de kube y la utilización de kubectl ,

```
613 mkdir -p $HOME/.kube  
614 sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  
615 sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Podemos comprobar desde el master si el node se ha unido de forma correcta

```
mark@torre-ubuntu:~$ kubectl get nodes  
NAME           STATUS    ROLES      AGE   VERSION  
node-kubernetes   Ready     <none>    26d   v1.23.5  
torre-ubuntu     Ready     control-plane,master  26d   v1.23.5
```

Me he encontrado con el fallo de la creación de la red interna con calico, lo he podido solucionar con otro (esto es para la conexión interna)

```
617 kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

El siguiente paso para asegurarme de su correcto funcionamiento fue poner un deployment a correr y ver su estado (no comentare aquí su creación porque fue una prueba básica de un servidor web y más adelante se explica detalladamente)

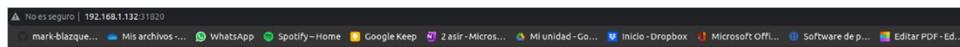
```
mark@torre-ubuntu:~$ kubectl get pod -o wide  
NAME          READY   STATUS    RESTARTS   AGE   IP           NODE   NOMINATED NODE   READINESS GATES  
nginx-8f567dbc-487xk  1/1    Running   1 (21h ago)  21h  10.224.1.13  node-kubernetes  <none>        <none>  
nginx-8f567dbc-74d42  1/1    Running   1 (21h ago)  21h  10.224.1.11  node-kubernetes  <none>        <none>  
nginx-8f567dbc-8s96d  1/1    Running   1 (21h ago)  21h  10.224.1.10  node-kubernetes  <none>        <none>  
nginx-8f567dbc-9rbt  1/1    Running   1 (21h ago)  21h  10.224.1.14  node-kubernetes  <none>        <none>  
nginx-8f567dbc-m7s6s  1/1    Running   1 (21h ago)  21h  10.224.1.9   node-kubernetes  <none>        <none>
```

Como podemos ver todos los pod se ejecutan en el node de kubernetes debido a razones de seguridad como esta página nos dice

<https://stackoverflow.com/questions/48729580/kubernetes-ds-wont-run-pod-on-master-node>

Pero como podemos ver podemos acceder a este servicio ya sea a través de la ip del node(slave) o del master de kubernetes y esto es importante porque cuando vayamos por dns apuntaremos al master

Master:



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

node:



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

En caso de querer ejecutar los pod en el master es posible como esa misma página web nos dice, en mi caso lo dejare en el node, ¡¡la seguridad ante todo!!

4. Desarrollo de la aplicación de encargos con react y node

La aplicación se divide en dos partes claras y diferenciadas.

Tenemos una parte visible o conocida como el front end que se ha hecho con react, que funciona por componentes que son reutilizables y como todos los framework supuestamente facilitan y agilizan el trabajo (no a mi parecer, prefiero html y javascript a pelo)

Esta parte es fácilmente visible porque es la aplicación que veremos sin más, formulario, footer header y al ser javascript podemos añadir funcionalidades como recorrer json , que son los objetos de javascript (parecidos a un array pero no es lo mismo), como dije en mi anteproyecto al ser esto sobre kubernetes toda la info k se almacene en estos pod son volátil y no podemos permitirnos ese lujo , como para la app de gestión de kubernetes tengo que crear un servidor web en nginx, es aquí donde se guardara este json(aun no lo tengo muy claro quizá sea en otro servidor node a parte , ya que aún no la he finalizado y este es uno de los puntos que me quedan) , al ir desarrollando esta aplicación solo con react me di cuenta que lo que pretendía que era que la app del pod recorriera el json ,lo mostrase y modificase ,me di cuenta que no se podía , este json debía de estar en el mismo proyecto, mis planes se arruinaron , y aparece la figura de un api rest.

Necesitaremos de node (con express, un framework, que simplemente quita un poco de código) el cual va a funcionar de servidor y en este se puede desarrollar el api que he mencionado, esto nos va a poder dar funcionalidades extra a nuestra app react y en concreto la que yo necesito que es añadir, deletear , y consultar info de un json hospedado en otros servidores (a traves de url)

Procedo a su explicación (dejo mi enlace a mi cuenta de github para que podáis observar el código si queréis <https://github.com/mark-blazquez/pag-kubernetes>)

4.1. Servidor back-end

Aquí se hallarán las rutas que el front end seguirá y otros datos y módulos utilizados por la aplicación incluso el puerto por el que este back end escuchará

```
//variables
const express = require('express');
const app = express();
const bodyParser = require('body-parser');
const morgan = require('morgan');
const path = require('path');
const req = require("express/lib/request");
const url = 'http://localhost:80/pedidos.json';
let options = { json: true };
const request = require('request');
//para escribir en un archivo
const fs = require('fs');
const json_pedidos = fs.readFileSync(path.join(__dirname , 'pedidos.json'), 'utf-8')
let pedidos = JSON.parse(json_pedidos)
var cors = require('cors')
//numero de puerto
app.set('port',8080);

//middleware
//formulario
app.use(cors());

//entender formularios
app.use(express.urlencoded({extended: false}));
//procesar datos json
app.use(express.json());
//morgan-para ver la respuestas desde consola
app.use(morgan('dev'))
app.set('view engine','ejs')
```

```
//creacion del servidor
app.listen(app.get('port'), ()=>{
    console.log ("servidor corriendo en el puerto 8080")
})
```

4.1.1. Ruta que obtiene los pedidos

```
// metodo get -devuelve el objeto pedidos a front end para que los muestre
app.get("/api", (req, res) => {
  //console.log(pedidos);
  res.send({pedidos: pedidos})
});
```

Lo único que hace es enviar un json al front end y este json lo obtiene de convertir un objeto (esto lo podéis ver en la imagen anterior a esta) que es el que está en local, pero más adelante deberá de estar en “nube”, json_pedidos es el objeto en bruto y pedidos es transformado a texto para poder trabajar con el

4.1.2. Ruta que guarda los pedidos (de un formulario a este archivo mencionado)

```
//metodo post -pasando un json local --añade el pedido enviado del front end a los pedidos globales
app.post('/api/nuevo',(req,res) => {
  //obtenemos el pedido del form
  //console.log(req.body)
  const newpedido = req.body
  //console.log(newpedido)
  //console.log(newpedido)
  //añadir el nuevo pedido al json obtenido del servidor
  pedidos.push(newpedido)
  //cojemos el json de epdidos y lo guardamos como texto
  const json_pedidos = JSON.stringify(pedidos)
  fs.writeFileSync(path.join(__dirname , 'pedidos.json'), json_pedidos,'utf-8')
  //ver el nuevo json con todos los pedidos
  //console.log(pedidos);
  res.redirect('/api/mid')
});
```

Se obtiene el pedido nuevo que viene como un objeto se puede ver en el req.body, con pedidos.push lo que hacemos es introducirlo al final del json definimos otra vez json_pedidos como los pedidos anteriores mas el nuevo como texto y con fs.write lo que hacemos es sobrescribir los datos anteriores en este documento .json y al final lo único que hace es redireccionar a una ruta que redirecciona al front end(ya que sino la página quedaría en blanco)

4.1.3. Ruta de borrado de pedidos

```
app.post('/api/delete',(req,res) => {
  //recibo el id del producto por formulario oculto
  const pedidoeliminar = req.body
  //aplico la función que define un objeto nuevo menos el pedido que concide con la id pasada
  pedidos = pedidos.filter(pedidos=> pedidos.id != pedidoeliminar.id )
  //console.log(pedidos);
  //se codifica para poder escribirse en el fichero
  const json_pedidos = JSON.stringify(pedidos)
  //se pasa al fichero para guardar el nuevo resultado
  fs.writeFileSync(path.join(__dirname , 'pedidos.json'), json_pedidos,'utf-8')
  res.redirect('/api/mid')
})
```

De igual forma recibimos el objeto de un formulario “invisible para el usuario” ya que el solo clicara un botón en pedido que quiere eliminar, pero pasa toda la info necesaria, entonces esto define ese objeto a eliminar, luego debemos de hacer el .filter que esto nada más y nada menos lo que hace es definir un objeto nuevo con todos los pedidos menos el que su id sea igual al id del objeto pasado por formulario y ya lo último hace lo mismo que lo anterior crea de nuevo el objeto json_pedidos para pasarlo como texto a este documento

Y hasta aquí lo que concierne a rutas con funciones ahora explicare otras cosas interesantes de rutas y para ello debo de explicar cómo va a ser mi aplicación, en un principio solo iba a ser la página de

gestión, pero para poder satisfacer los deseos de aquellos que quieran ver una página con un diseño más elaborado voy a realizar una página que será la que el usuario acceda, de ahí esta ruta

4.1.4. Index

```
app.get("/", (req, res) => {
  //console.log(pedidos);
  //res.sendFile(path.resolve(__dirname, '../asador/build', 'index.html'));
  res.redirect('http://personal.com:80')
});
```

Esto lo que hace es mostrar la página que se haya en el puerto 80 y esto necesita de una explicación y más adelante la voy a dar , simplemente necesitamos saber ahora mismo que esta página tendrá un inicio de sesión a través de Google y que solo permitirá acceso a mi usuario , a través de código , ya que si lo enfocamos en un ambiente real solo queremos que acceda el gestor de la tienda y no cualquier usuario con una cuenta de Google, y en caso de ser otro usuario volcara un error por pantalla , para esto he utilizado el módulo Passports

4.1.5. Ruta login Google

```
//inicio de sesión con google
var passport = require('passport');
var GoogleStrategy = require('passport-google-oauth2').Strategy;
const {Claves} = require('../claves/claves');
const { redirect } = require('../express/lib/response');

passport.serializeUser(function(user, done) {
  done(null, user);
});
passport.deserializeUser(function(user, done) {
  done(null, user);
});
let usuario="";
passport.use(
  new GoogleStrategy(
    {
      callbackURL: '/auth/google/callback',
      clientID: Claves.client_id,
      clientSecret: Claves.client_secret,
    },
    (accessToken, refreshToken, profile, cb) => {
      //console.log(accessToken)
      //console.log(profile)
      if(profile._json.email === "blazquezmark97@gmail.com"){
        //console.log("correct user")
        return cb(null, profile);
      }else{
        // fail
        //console.log("bad user")
        //return usuario = "erroneo"
        cb(new Error("no eres el ususario adecuado!"));
      }
    }
  )
)
//google passport routes
app.get('/api/auth/google',
  passport.authenticate('google', {
    scope:
      ['email', 'profile']
  })
);
app.get('/auth/google/callback',
  passport.authenticate('google', {
    successRedirect: '/api/mid',
    failureRedirect: '/'
  })
);
```

El botón apunta a la ruta /api/auth/google y esta llama al método authenticate que utiliza la googlestrategy que es de aquí de donde se obtiene el scope necesario y una vez que lo completa llama al callback que lo único que hace es que si esta verificación ha sido correcta envía a la ruta que te devuelve a la página de pedidos como podéis ver a continuación

4.1.6. Ruta redirección a página pedidos

```
  ,
  app.get("/api/mid", (req, res) => {
    res.redirect('http://personal.com:81')
  });
}
```

La validación del usuario la hace porque esta identificación que hace a través de Google obtiene un objeto llamado profile y en este se encuentra datos como nombre, foto ,etc, etc , un objeto dentro de este objeto es _json y dentro de este email entonces si lo que hacemos es requerir que este sea el email deseado pasa a la siguiente sino vuelca el error ..

Lo que se ve de claves.loksea. es porque como es una aplicación que va a usar el api de Google debemos de darnos de alta en la consola de Google y se nos darán unas credenciales que nos certifican como que es una aplicación real y permite esta consulta al api de Google, aquí una muestra de la consola mencionada

The screenshot shows the Google Cloud Platform interface for managing APIs and services. On the left, there's a sidebar with options like 'API y servicios', 'Biblioteca', 'Credenciales', 'Pantalla de consentimiento de OAuth', 'Verificación del dominio', and 'Acuerdos de uso de páginas'. The 'Pantalla de consentimiento de OAuth' option is highlighted. The main area is titled 'proyecto final' and shows the 'Estado de publicación' as 'Prueba'. There's a button labeled 'PUBLICAR LA APLICACIÓN'. Below this, there's a section titled 'Tipo de usuario'.

4.1.7. Ruta de cerrar sesión

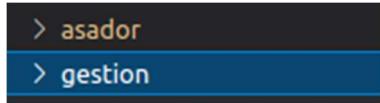
```
//cerrar session
app.get("/logout", (req, res) => {
  //hay q cerrar sesion
  //redireccionamos a index
  res.redirect('/')
});
```

Nos re direcciona al index y cierra sesión

4.2. Front end

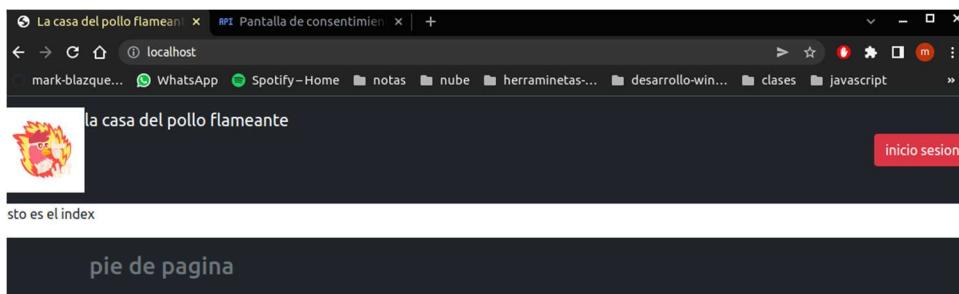
Ahora pasare a mostrar el front end y como decía anteriormente necesito hacer una aclaración previa react es un framework complejo y dispongo de poco tiempo, así que en mi búsqueda de realizar diferentes paginas osea index, indexlogueao...etc no encontré la formula todo lo que obtenía era como mostrar en la misma pag diferentes partes (con un módulo enrutador)

A parte cuando está en desarrollo dispone de un servidor propio que cuando se lleva a producción no debemos de utilizar, es por ello que lo que he tenido que hacer es dos proyectos de react y estos se deben de compilar que es lo que nos dará una página estática que es lo que mostrara el servidor web



Uno es para cuando se está logueado y otro para cuando no, que lo que me permite es tener paginas distintas que deben de estar hospedados en diferentes puertos es por ello que uno sera el 80 y otro el 81, que quizá cambie cuando meta los certificados, o no

4.2.1. Index



Esta será la página de inicio evidentemente no está terminada y se puede ver el botón de Google pasamos a ver el código

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      | name="description"
      | content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" />
    <!--
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the 'public' folder during the build.
      Only files inside the 'public' folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <title>La casa del pollo flameante</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>

```

Como he dicho es una página, y en esta se encuentra el div root que es donde ocurre toda la magia ya que parece que no tiene nada, pero si

```

import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';

import reportWebVitals from './reportWebVitals';

ReactDOM.render(
  <React.StrictMode>
    |   <App/>
  </React.StrictMode>,
  document.getElementById('root')
);

// If you want to start measuring performance in your app, pass
// to log results (for example: reportWebVitals(console.log()))
// or send to an analytics endpoint. Learn more: https://bit.ly/reportWebVitals\(\)

```

Aquí se puede ver como en este div llamado root se renderiza algo llamado app ¿y que es app? Pues el siguiente fragmento de código nos lo dice

```
import "bootstrap/dist/css/bootstrap.min.css";
import React, { Component } from "react";
import Cabecera from './componentes/cabecera';
import Pie from './componentes/pie';
import CuerpoIndex from './componentes/cuerpoindex';

function App() {
    return (
        <div>
            <Cabecera />
            <CuerpoIndex/>
            <Pie />
        </div>
    );
}

export default App;
```

Como mencione al principio react va por componentes y aquí se puede ver bien claro como llamo a diferentes partes que hacen el todo, esta página tiene poco de interesante ya que solo tiene 1 funcionalidad la del inicio de sesión así que lo que mostrare es parte de código que es la cabecera

```
import React, { Component } from "react";
import logo from './Logo.jpg';
class Cabecera extends React.Component {
  render(){
    return(
      <nav className="navbar navbar-expand-lg navbar-dark bg-dark">
        <a className="navbar-brand" >
          |   <img src={logo} className="d-inline-block align-top" width="100" height="100" ></img>la casa del pollo flameante
          |</a>

        <div className="collapse navbar-collapse" id="navbarSupportedContent">
          <ul className="navbar-nav mr-auto">
            <li className="nav-item active">
              |   <a href="#">Inicio</a>
            </li>
            <li className="nav-item">
              |   <a href="#">Sobre Nosotros</a>
            </li>
          </ul>
        </div>

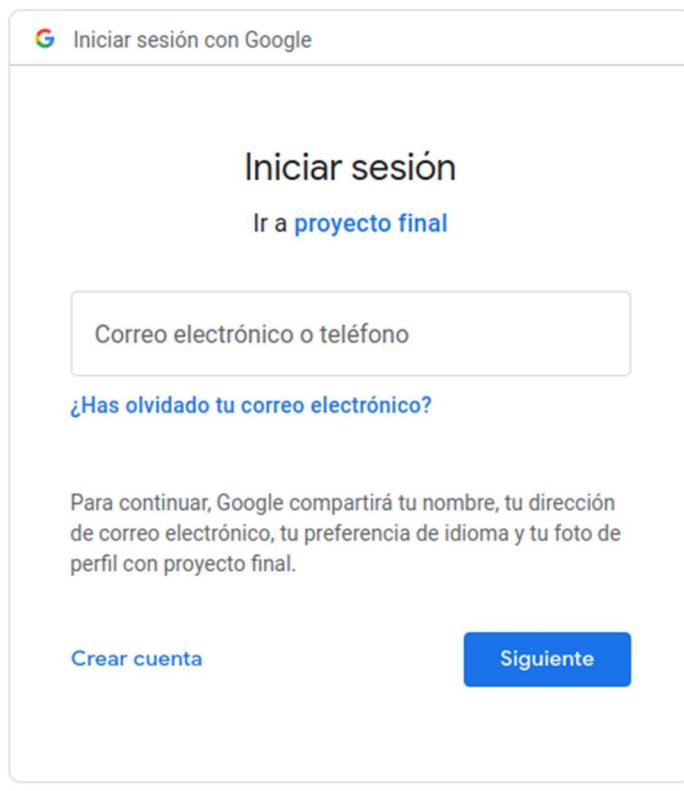
        <div>
          |   <a className="btn btn-danger" href="http://personal.com:8080/api/auth/google">inicio sesion </a>
        </div>
      </nav>
    )
  }
}

export default Cabecera;
```

Podemos ver que el botón de google no es nada más que un href a nuestro servidor node y en específico a la ruta que se encarga de hacer este inicio de sesión así que si pinchamos en él y por supuesto con nuestro servidor node corriendo de fondo podemos observar lo siguiente

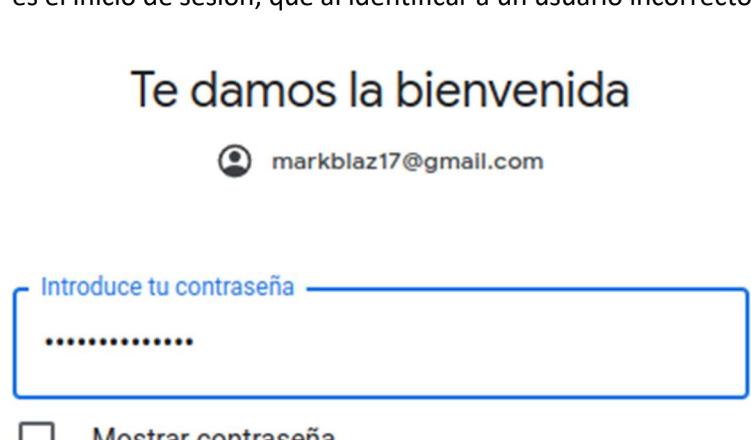
```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ npm run dev
> react-node@1.0.0 dev
> nodemon src/index.js morgan

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node src/index.js morgan`
servidor corriendo en el puerto 8080
GET /api/auth/google 302 6.681 ms - 0
□
```



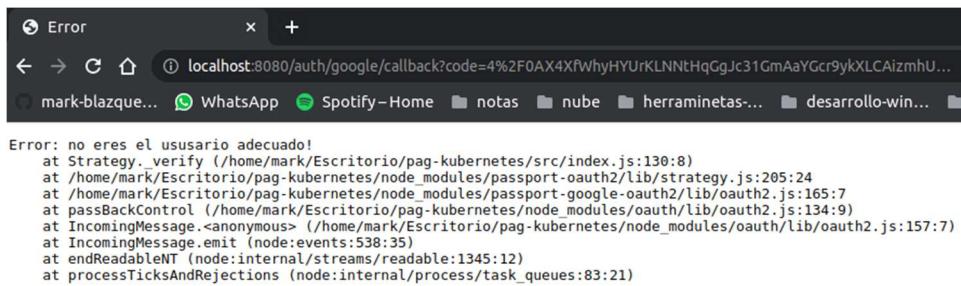
The image shows a screenshot of a Google sign-in page. At the top, there's a header with the Google logo and the text "Iniciar sesión con Google". Below the header, the main title is "Iniciar sesión". There's a blue link "Ir a proyecto final". A large input field is labeled "Correo electrónico o teléfono". Below the input field, a link says "¿Has olvidado tu correo electrónico?". A note states: "Para continuar, Google compartirá tu nombre, tu dirección de correo electrónico, tu preferencia de idioma y tu foto de perfil con proyecto final." At the bottom, there are two buttons: "Crear cuenta" and a larger blue "Siguiente" button. At the very bottom of the page, there are links for "Español (España) ▾", "Ayuda", "Privacidad", and "Términos".

Y es el inicio de sesión, que al identificar a un usuario incorrecto nos volcara el error



The image shows a screenshot of a Google sign-in error page. The main heading is "Te damos la bienvenida". Below it, there's a user profile icon and the email address "markblaz17@gmail.com". A large input field is labeled "Introduce tu contraseña" with a placeholder ".....". Below the input field is a checkbox labeled "Mostrar contraseña".

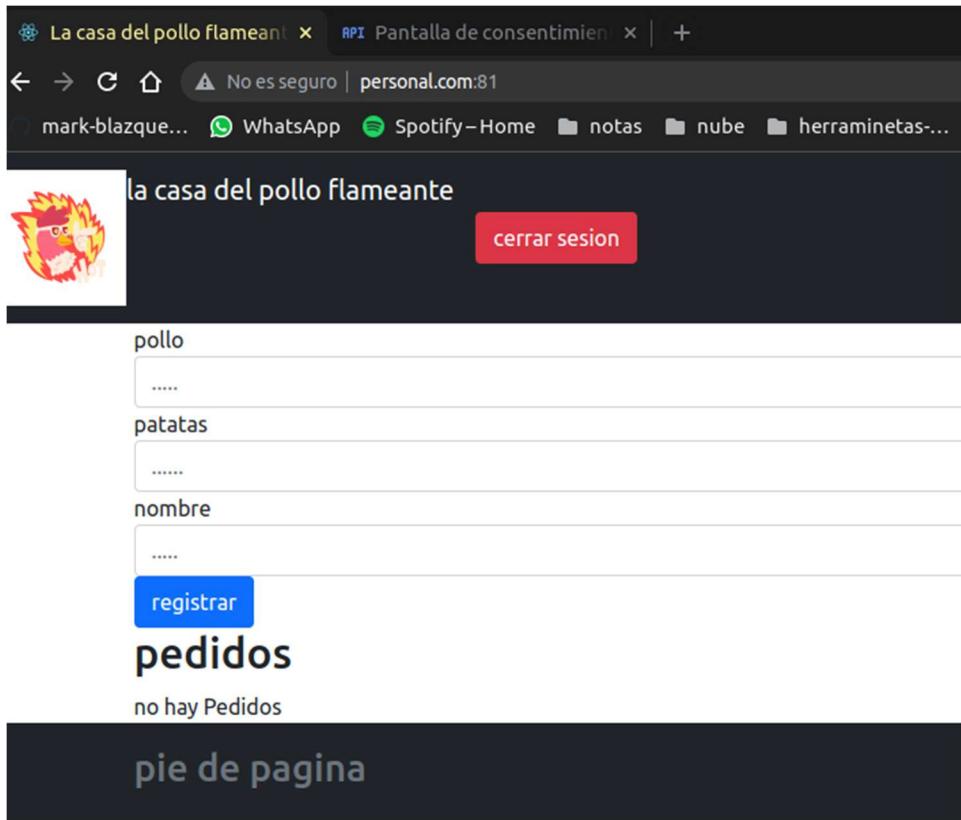
Recordemos que el único usuario que tiene acceso es el blazquezmark97@.....



```
Error: no eres el usuario adecuado!
at Strategy.verify (/home/mark/Escritorio/pag-kubernetes/src/index.js:130:8)
at /home/mark/Escritorio/pag-kubernetes/node_modules/passport-oauth2/lib/strategy.js:205:24
at /home/mark/Escritorio/pag-kubernetes/node_modules/passport-google-oauth2/lib/oauth2.js:165:7
at passBackControl (/home/mark/Escritorio/pag-kubernetes/node_modules/oauth/lib/oauth2.js:134:9)
at IncomingMessage.<anonymous> (/home/mark/Escritorio/pag-kubernetes/node_modules/oauth/lib/oauth2.js:157:7)
at IncomingMessage.emit (node:events:538:35)
at endReadableNT (node:internal/streams/readable:1345:12)
at processTicksAndRejections (node:internal/process/task_queues:83:21)
```

En cambio, cuando el usuario es el correcto nos redirecciona a

4.2.2. Página pedidos



Que como se puede ver va por el puerto 81 y por dns(que en realidad es el /etc/hosts para que fuese el mismo el docker que en mi pc)

Una vez aquí muestro su estructura

```

import "bootstrap/dist/css/bootstrap.min.css";
import React, { Component } from "react";
import Cabecera from './componentes/cabecera';
import Formulario from './componentes/formulario';
import Pedidos from './componentes/pedidos';
import Pie from './componentes/pie';

function App() {

    return (
        <div>
            <Cabecera />
            <Formulario />
            <Pedidos />
            <Pie />
        </div>
    );
}

export default App;

```

Aquí me voy a centrar en lo que de verdad importa que es formulario y pedidos ya que la cabecera hace lo mismo que la cabecera de cuando no se está logueado a excepción de que la ruta es diferente y nos lleva a cerrar sesión

```

import React, { Component } from "react";
import logo from './logo.jpg';
class Cabecera extends React.Component {
    render(){
        return(
            <nav className="navbar navbar-expand-lg navbar-dark bg-dark">
                <a className="navbar-brand " href="#">
                    <img src={logo} className="d-inline-block align-top" width="100" height="100" />la casa del pollo
                </a>
                <button className="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent"
                    <span className="navbar-toggler-icon"></span>
                </button>

                <div>
                    <a className="btn btn-danger " href="http://personal.com:8080/logout">cerrar sesion </a>
                </div>
            </nav>
        )
    }
}
export default Cabecera;

```

1.1.1. Formulario

pollo
.....
patatas
.....
nombre
.....
registrar

Esta parte es el formulario que nos permite introducir nuevos pedidos en el json como ya explicado antes a través de un objeto que crea y tal ... es hora de visualizarlo

```

render() {
  return (
    <form className="container" method="POST" action="http://personal.com:8080/api/nuevo" name="formulario">
      <div className="form-group">
        <label>pollo</label>
        <input type="number" className="form-control" name="pollo" placeholder="....." />
      </div>
      <div className="form-group">
        <label>patatas</label>
        <input type="number" className="form-control" name="patatas" placeholder="....." />
      </div>
      <div className="form-group">
        <label>nombre</label>
        <input type="text" className="form-control" name="nombre" placeholder="....." />
      </div>
      <div className="form-group d-none"> /*lo que hace es crear un campo oculto con el valor aleatorio definido arriba y pa
        <label>id</label>
        <input type="number" className="form-control" name="id" value={this.state.id} />
      </div>
      <button type="submit" className="btn btn-primary">registrar</button>
    </form>
  )
}

```

A simple vista es un formulario normal, pero debemos de tener muy en cuenta los nombre y el campo value del id y que va por post a la ruta node debida

```

import React, { Component } from "react";
class Formulario extends React.Component {
  //creamos el objeto que vamos a pasar a la api, aqui vacio
  constructor(props) {
    super(props);
    this.state = {
      pollo:"",
      patatas:"",
      nombre:"",
      id: Math.random()//esto hace un id aleatorio
    }
    this.handleInputChange = this.handleInputChange.bind(this);
  }
  //definimos constantes que van a tener el valor del formulario
  handleInputChange(event) {
    const target = event.target;
    const pollo = target.pollo;
    const patatas = target.patatas;
    const nombre = target.nombre;
    const id = target.id;

    //creamos el objetal que vamos a pasar a la api pero esta vez con sus valores recuperados del form

    this.setState({
      pollo: pollo,
      patatas: patatas,
      nombre: nombre,
      id: id
    });
  }
}

```

Esta porción de código es la interesante ya que es donde se ve como en primer lugar creamos un objeto vacíos , menos id que es una función matemática que nos da un numero entre 0 y 1 con muchos decimales, luego podemos ver como la función inputchange lo que hace es definir las constantes con el valor que tenga en el formulario y por ultimo crea ese objeto que tiene el valor de esas constantes , recordar que el id no lo define el formulario , ya que es un campo invisible por su propiedad d-none sino que viene dado por el campo value del formulario que es igual al valor del objeto id

Veamos su funcionalidad

The form consists of three text input fields and one button:

- pollo:** Value: 1
- patatas:** Value: 1
- nombre:** Value: mark

Registrar

Veamos el resultado en el json

```
js index.js      {} pedidos.json M ×
src > {} pedidos.json > ...
1  [{"pollo": "1", "patatas": "1", "nombre": "mark", "id": "0.22376169809271662"}]
```

Lo del id lo tenemos que tener en cuenta porque si borrásemos por nombre podríamos borrar otro de mismo nombre (que en Andalucía todos o se llaman juan, pepe, Antonio, maría, Carmen, y García o López de apellido) y si fuese un id auto incremental por posición en el json podría dar error por ejemplo si tenemos 5 y se borra el 4 el nuevo encargo ocuparía la posición 5 y ahí ya surgiría conflicto porque ya hay un 5 y cuando se borre este borraría los dos encargos con id 5

Entonces tenemos que ver el json para saber nuestros pedidos así que veamos lo que el usuario

1.1.2. Muestra de pedido y botón de borrado

pedidos

pollos 1 patatas 1 nombre mark borrar

Y ahora el código

```
renderPedidos(){
  if (this.state_pedidos.length<=0){
    return <div>no hay Pedidos</div>
  }else{
    return this.state_pedidos.map((val,key)>{
      return (
        <div className="d-flex" key={key}>
          <div>
            <span>pollos </span>{val.pollo} <span> patatas </span> {val.patatas}<span> nombre </span>{val.nombre}
          </div>
          <div>{/*lo que hace es crear un campo oculto con el valor aleatorio definido arriba y pasarlo por formulario*/}
            <form className="d-flex" method="POST" action="http://personal.com:8080/api/delete" name="google">
              <div className="form-group d-none">
                <label>pollo</label>
                <input type="number" className="form-control" name="pollo" value={val.pollo}/>
              </div>
              <div className="form-group d-none">
                <label>patatas</label>
                <input type="number" className="form-control" name="patatas" value={val.patatas} />
              </div>
              <div className="form-group d-none">
                <label>nombre</label>
                <input type="text" className="form-control" name="nombre" value={val.nombre}/>
              </div>
              <div className="form-group d-none">
                <label>id</label>
                <input type="number" className="form-control" name="id" value={val.id} />
              </div>
              <button type="submit" className="btn btn-primary" >borrar</button>
            </form>
          </div>
        </div>
      )
    })
  }
}
```

```
render(){
  return (
    <div className="container">
      <h2>pedidos</h2>
      {this.renderPedidos()}
    </div>
  )
}
```

Esta parte se hace con una función y el porque es por la comprobación, en caso de no tener pedidos mostrar no hay pedido en caso de tener lo que hace es mostrar el valor de este pedido

Es importante a mencionar que crea el formulario de eliminado con los datos de este pedido que es oculto para el usuario a excepción del botón de submit , pero ¿de dónde sacamos esos datos? pues veámoslo

```
class Pedidos extends React.Component {  
  constructor(props){  
    super(props);  
    this.state={  
      pedidos:[],  
      pollo:"",  
      patatas:"",  
      nombre:"",  
      id: ""  
    }  
    this.handleInputChange = this.handleInputChange.bind(this);  
  }  
  
  componentDidMount() {  
    fetch('http://personal.com:8080/api')  
      .then(response => response.json())  
      .then(res =>{  
        this.setState({pedidos: [...this.state.pedidos, ...res.pedidos]})  
      })  
  }  
  
  handleInputChange(event) {  
    const target = event.target;  
    const pollo = target.pollo;  
    const patatas = target.patatas;  
    const nombre = target.nombre;  
    const id = target.id;  
  
    //creamos el objetal que vamos a pasar a la api pero esta vez con sus valores recuperados del form  
  
    this.setState({  
      pollo: pollo,  
      patatas: patatas,  
      nombre: nombre,  
      id: id  
    })  
  }  
}
```

Lo que hace es recorrer el objeto que manda el back end que lo hacia la ruta /api se puede ver en el fech, y la otra parte del código hace igual que lo anteriormente visto definir un objeto que pasara el form a la ruta post /api/delete pero esta vez lo que tiene el formulario son los valores dado por este objeto que ya hemos obtenido

Así que voy a meter dos pedidos más y borrar el del medio(de los chichos) para ver que funciona

pedidos

pollos 1 patatas 1 nombre mark

borrar
borrar
borrar

pollos 2 patatas 2 nombre pepe

pollos 3 patatas 3 nombre pepa

pedidos

pollos 1 patatas 1 nombre mark

borrar
borrar

pollos 3 patatas 3 nombre pepa

Para evitar el recargo de página manual, inserto un timer que cada segundo refresque el componente que obtiene la información de los pedidos de esa forma evito la perdida de informacion pensando de cara a produccion

```
componentDidMount() {
  //funcion para hacer la peticion de forma recursiva a node para los cambios en el json solo recargando esa parte del codigo
  setInterval(() => {
    fetch('https://torre-ubuntu.ddns.net:31059/api/muestra')
      .then(response => response.json())
      .then(res =>{
        //borra el objeto definido anteriormente para que no lo imprima en el front end
        this.setState({pedidos:[] })
        this.setState({pedidos: [...this.state.pedidos, ...res.pedidos] })
      })
  }), 1000);
}
```

5. Creación del contenedor docker

5.1. Servidor web

Partiremos de la imagen nginx que nos proporciona dockerhub y crearemos un contenedor para crear la imagen personalizada que kubernetes utilizará y después de varias formas intentadas la mejor es a partir de un Dockerfile ya que si no me provoca un fallo en kubernetes , y lo que haremos es pasarle los compilados de react a este contenedor, al directorio donde esta imagen ofrece el contenido web

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ cd gestion/
mark@torre-ubuntu:~/Escritorio/pag-kubernetes/gestion$ npm run build

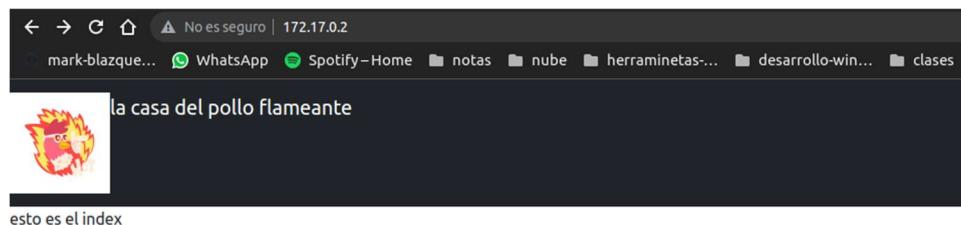
> gestion@0.1.0 build
> react-scripts build
```

```
FROM nginx:stable-alpine
COPY ./asaror/build/ /usr/share/nginx/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile -t markblazquez/contfront:v11 .
Sending build context to Docker daemon 728.1MB
Step 1/4 : FROM nginx:stable-alpine
--> 5c05ca045835
Step 2/4 : COPY ./asaror/build/ /usr/share/nginx/html
--> 793eeb16163a
Step 3/4 : EXPOSE 80
--> Running in c373cc94e126
Removing intermediate container c373cc94e126
--> a63a6483608d
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
--> Running in c374fad8cdc0
Removing intermediate container c374fad8cdc0
--> 24349e35d223
Successfully built 24349e35d223
Successfully tagged markblazquez/contfront:v11
```

Con esto ya podemos probar su funcionamiento en el contenedor desde nuestra maquina anfitriona

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker run -it --rm -p 8787:80 markblazquez/contfront:v11
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2022/05/02 17:40:00 [notice] 1#1: using the "epoll" event method
2022/05/02 17:40:00 [notice] 1#1: nginx/1.20.2
2022/05/02 17:40:00 [notice] 1#1: built by gcc 10.3.1 20210424 (Alpine 10.3.1_git20210424)
2022/05/02 17:40:00 [notice] 1#1: OS: Linux 5.13.0-40-generic
```



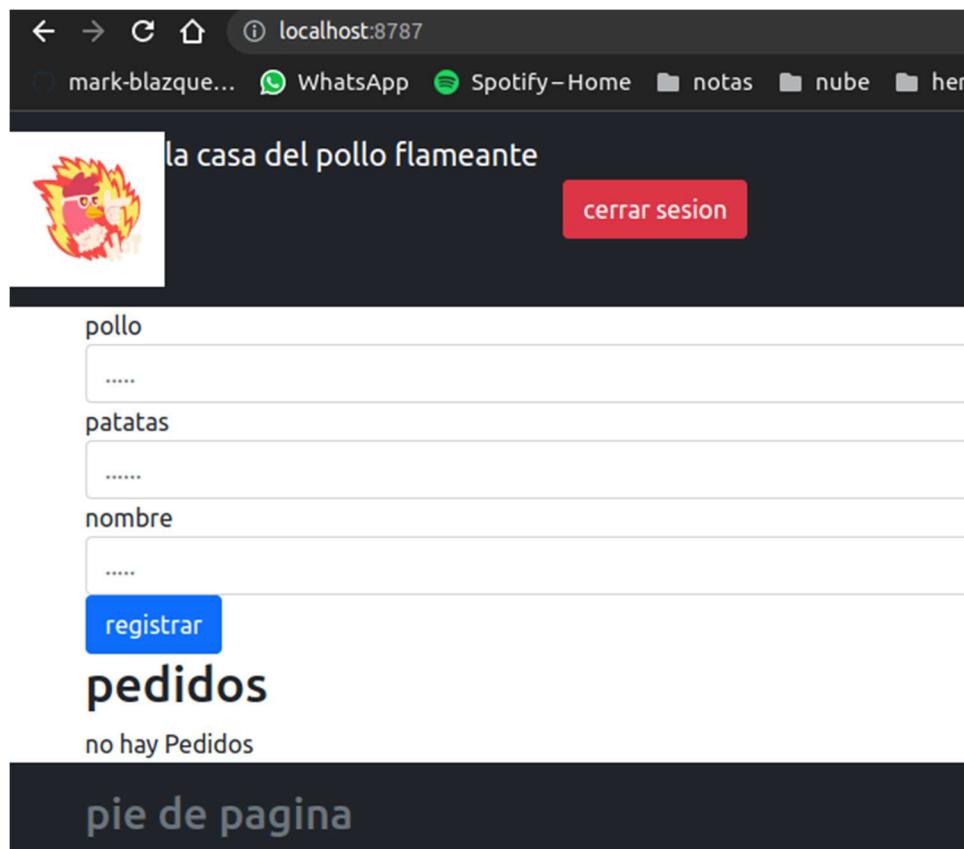
pie de pagina

Voy a separar las dos partes de la app en diferentes contenedores para hacer deployment diferentes en kubernetes así que hago otro dockerfile para la parte de gestión

```

mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile2 -t markblazquez/contfront:v12 .
Sending build context to Docker daemon 728.1MB
Step 1/4 : FROM nginx:stable-alpine
--> 5c05ca045835
Step 2/4 : COPY ./gestion/build/ /usr/share/nginx/html
--> 6ee7a554de2a
Step 3/4 : EXPOSE 80
--> Running in a885f9a9b6a3
Removing intermediate container a885f9a9b6a3
--> b45026c772ba
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
--> Running in 4c4bfdc7ec5f
Removing intermediate container 4c4bfdc7ec5f
--> 8d7bb412296b
Successfully built 8d7bb412296b
Successfully tagged markblazquez/contfront:v12
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker run -it --rm -p 8787:80 markblazquez/contfront
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2022/05/02 18:07:46 [notice] 1#1: using the "epoll" event method
2022/05/02 18:07:46 [notice] 1#1: nginx/1.20.2
2022/05/02 18:07:46 [notice] 1#1: built by gcc 10.3.1 20210424 (Alpine 10.3.1_git20210424)
2022/05/02 18:07:46 [notice] 1#1: OS: Linux 5.13.0-40-generic
2022/05/02 18:07:46 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2022/05/02 18:07:46 [notice] 1#1: start worker processes
2022/05/02 18:07:46 [notice] 1#1: start worker process 33
2022/05/02 18:07:46 [notice] 1#1: start worker process 34
2022/05/02 18:07:46 [notice] 1#1: start worker process 35
2022/05/02 18:07:46 [notice] 1#1: start worker process 36
2022/05/02 18:07:46 [notice] 1#1: start worker process 37
2022/05/02 18:07:46 [notice] 1#1: start worker process 38
2022/05/02 18:07:46 [notice] 1#1: start worker process 39
2022/05/02 18:07:46 [notice] 1#1: start worker process 40
172.17.0.1 - - [02/May/2022:18:08:17 +0000] "GET / HTTP/1.1" 200 620 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML like Gecko) Chrome/100.0.4896.60 Safari/537.36" "-"
172.17.0.1 - - [02/May/2022:18:08:17 +0000] "GET /static/js/main.490ebad9.js HTTP/1.1" 200 150617 "
ecko) Chrome/100.0.4896.60 Safari/537.36" "-"
172.17.0.1 - - [02/May/2022:18:08:17 +0000] "GET /static/css/main.38be36b0.css HTTP/1.1" 200 166614 "

```



Ambas funcionan así que es hora de crear la imagen que ira en kubernetes así que hacemos el push y el pull para asegurarnos que no nos da un fallo al no encontrar la imagen especificada luego en kubernetes

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker push markblazquez/contfront:v11
The push refers to repository [docker.io/markblazquez/contfront]
404478b9dbcc: Pushed
cdb3cabb76c7: Mounted from library/nginx
badc52910423: Mounted from library/nginx
b20e4b98292c: Mounted from library/nginx
02a247a96218: Mounted from library/nginx
12e3caaf0146: Mounted from library/nginx
b541d28bf3b4: Mounted from library/nginx
V11: digest: sha256:c3f0258165ece58649f0032f3e7c5d41e05c6f5eaa339e7d166295d405dfd02e size: 1778
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker pull markblazquez/contfront:v11
V11: Pulling from markblazquez/contfront
Digest: sha256:c3f0258165ece58649f0032f3e7c5d41e05c6f5eaa339e7d166295d405dfd02e
Status: Image is up to date for markblazquez/contfront:v11
docker.io/markblazquez/contfront:v11
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ ssh mark@node
mark@node's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-40-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Se pueden aplicar 7 actualizaciones de forma inmediata.
Para ver estas actualizaciones adicionales ejecute: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Mon May  2 19:26:57 2022 from 192.168.1.132
mark@node-kubernetes:~$ sudo docker pull markblazquez/contfront:v11
[sudo] contraseña para mark:
V11: Pulling from markblazquez/contfront
Digest: sha256:c3f0258165ece58649f0032f3e7c5d41e05c6f5eaa339e7d166295d405dfd02e
```

5.2. Back end

Comenzamos con la creación del contenedor

```
mark@torre-ubuntu:~$ docker run -itd --name backend debian bash  
134a03b1461c2f0fc6f459b3c715a2b51514898e2e2c08bb39deb30aa3827cf  
mark@torre-ubuntu:~$ docker exec -it backend bash
```

Descargamos el proyecto

```
root@134a03b1461c:/var/pag-kubernetes# ls  
node_modules  package-lock.json  package.json  src
```

Instalamos node y npm

```
root@134a03b1461c:/var/pag-kubernetes# apt install nodejs  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  
Processing triggers for libc-bin (2.31-15ubuntu10.2) ...  
root@134a03b1461c:/var/pag-kubernetes# apt install npm  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  adwaita-icon-theme at-spi2-core binutils binutils-common b
```

Instalamos los módulos que el proyecto necesita para funcionar y que se hayan en package.json

```
root@134a03b1461c:/var/pag-kubernetes# npm install package.json  
npm WARN deprecated package.json@2.0.1: Use pkg.json instead.  
  
added 81 packages, and audited 308 packages in 11s
```

Y creamos el archivo con las credenciales de Google que no está en GitHub por seguridad

```
root@134a03b1461c:/var/pag-kubernetes/src/claves# nano claves.js  
root@134a03b1461c:/var/pag-kubernetes/src/claves#
```

Comprobamos que funciona

```
root@134a03b1461c:/var/pag-kubernetes/src# node index.js  
servidor corriendo en el puerto 8080
```



Con esto ya sabemos que funciona así que lo que queda es guardar el contenedor crear una imagen y a partir de esta vamos a crear una imagen que es la que utilizará Kubernetes pero esta debe de tener por defecto activado el servidor node de inicio y eso lo vamos a hacer con un Dockerfile

```
mark@torre-ubuntu:~/Escritorio/kubernetes$ docker commit backend markblazquez/backendk  
sha256:affe0a6a5e96b4a34e792c0c48915b31c34acdf42d36e10c82c345b652ab109e
```

```
FROM markblazquez/backendk  
  
# Create app directory  
WORKDIR /var/pag-kubernetes/  
  
# Bundle app source  
  
EXPOSE 8080  
  
CMD [ "node", "src/index.js" ]
```

Y se crea la imagen

```
mark@torre-ubuntu:~/Escritorio/kubernetes$ docker build --tag markblazquez/backendk:v2 .  
Sending build context to Docker daemon 70.14kB  
Step 1/4 : FROM markblazquez/backendk  
--> affe0a6a5e96  
Step 2/4 : WORKDIR /var/pag-kubernetes/  
--> Running in 48686e542559  
Removing intermediate container 48686e542559  
--> f65889115a00  
Step 3/4 : EXPOSE 8080  
--> Running in 73d35c2f1726  
Removing intermediate container 73d35c2f1726  
--> ec6aceee4a52  
Step 4/4 : CMD [ "node", "src/index.js" ]  
--> Running in 143daadbc289  
Removing intermediate container 143daadbc289  
--> d223f4e984bc  
Successfully built d223f4e984bc  
Successfully tagged markblazquez/backendk:v2
```

6. Implantación de la aplicación en kubernetes

6.1. Front end

Partiremos del archivo yaml que utilice en la prueba del servidor web cuando instale kubernetes en forma de cluster pero ahora adaptare este a mis necesidades, en este caso será cambiar la imagen que los pods utilizaran que será la que acabo de crear en docker

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
  labels:
    app: nginx
    color: green
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
        color: green
    spec:
      containers:
        - name: nginx
          image: markblazquez/confront:v11
          imagePullPolicy: IfNotPresent
          ports:
            - name: uno
              protocol: TCP
              containerPort: 80
            - name: dos
              protocol: TCP
              containerPort: 81
#
#           ports:
#             - name: uno
#               port: 80
#               targetPort: 80
#               protocol: TCP
#             - name: dos
#               port: 81
#               targetPort: 9081
#               protocol: TCP
#           selector:
#             app: nginx
#           sessionAffinity: None
#           type: NodePort
```

Aplicamos los cambios y vemos los resultados

```
mark@torre-ubuntu:~/Escritorio/kubernetes$ kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx configured
service/nginx-demo-service unchanged
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ kubectl get all
NAME                           READY   STATUS    RESTARTS   AGE
pod/nginx-dd567d6bd-gd9rg     1/1    Running   0          2m38s
pod/nginx-dd567d6bd-mq596     1/1    Running   0          56s
pod/node-64cc6c867-r7c5p     1/1    Running   0          31h

NAME                  TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1   <none>        443/TCP   34d
service/nginx-demo-service   NodePort    10.107.134.113 <none>        80:31952/TCP   2m38s
service/node-demo       NodePort    10.108.18.16  <none>        8080:31059/TCP   6d23h

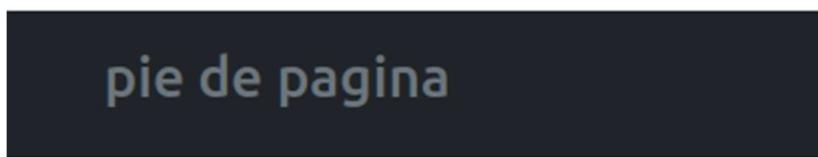
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx 2/2     2           2           16m
deployment.apps/node   1/1     1           1           6d23h

NAME                DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-6957d5f965 0         0         0       16m
replicaset.apps/nginx-dd567d6bd 2         2         2       2m38s
replicaset.apps/node-64cc6c867 1         1         1       6d22h
```

Y observamos que podemos acceder a través de nuestra maquina por el puerto de kubernetes



esto es el index



Y para la página de gestión igual

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx2
  labels:
    app: nginx2
    color: green
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx2
  template:
    metadata:
      labels:
        app: nginx2
        color: green
    spec:
      containers:
        - name: nginx2
          image: markblazquez/contfront:v12
          imagePullPolicy: IfNotPresent
          ports:
            - name: uno
              protocol: TCP
              containerPort: 80
#            - name: dos
#              protocol: TCP
#              containerPort: 81
#
apiVersion: v1
kind: Service
metadata:
  annotations:
  name: nginx-demo-service2
spec:
  ports:
    - name: uno
      port: 80
      targetPort: 80
      protocol: TCP
#    - name: dos
#      port: 81
#      targetPort: 9081
#      protocol: TCP
  selector:
    app: nginx2
  sessionAffinity: None
  type: NodePort

mark@torre-ubuntu:~/Escritorio/kubernetes$ kubectl apply -f ./kubernetes/nginx-deployment2.yaml
deployment.apps/nginx2 unchanged
service/nginx-demo-service2 configured
```

```
mark@torre-ubuntu:~/Escritorio/kubernetes$ kubectl get all
NAME                               READY   STATUS    RESTARTS   AGE
pod/nginx-dd567d6bd-gd9rg        1/1     Running   0          53m
pod/nginx-dd567d6bd-mq596        1/1     Running   0          52m
pod/nginx2-8469bf47c5-bl45d      1/1     Running   0          18m
pod/nginx2-8469bf47c5-tv2wn      1/1     Running   0          18m
pod/node-64cc6c867-r7c5p         1/1     Running   0          32h

NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
service/kubernetes   ClusterIP   10.96.0.1      <none>           443/TCP       34d
service/nginx-demo-service   NodePort    10.107.134.113  <none>           80:31952/TCP  53m
service/nginx-demo-service2  NodePort    10.109.151.31  <none>           80:31317/TCP  18m
service/node-demo      NodePort    10.108.18.16   <none>           8080:31059/TCP 7d

NAME                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx   2/2      2           2           67m
deployment.apps/nginx2  2/2      2           2           18m
deployment.apps/node    1/1      1           1           7d

NAME            DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-6957d5f965 0          0          0          67m
replicaset.apps/nginx-dd567d6bd  2          2          2          53m
replicaset.apps/nginx2-8469bf47c5 2          2          2          18m
replicaset.apps/node-64cc6c867   1          1          1          6d23h
```

← → C ⌂ No es seguro | 10.109.151.31

mark-blazque... WhatsApp Spotify – Home notas nube herramientas-...

 la casa del pollo flameante cerrar sesión

correo

.....

contraseña

.....

nombre

.....

registrar

pedidos

no hay Pedidos

pie de pagina

6.2. Back end

Creamos un deployment con el puerto especifico de node

```
---  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: node  
  labels:  
    app: node  
    color: green  
spec:  
  replicas: 2  
  selector:  
    matchLabels:  
      app: node  
  template:  
    metadata:  
      labels:  
        app: node  
        color: green  
    spec:  
      containers:  
        - name: node  
          image: markblazquez/backendk:v2  
          imagePullPolicy: IfNotPresent  
          ports:  
            - name: node  
              protocol: TCP  
              containerPort: 8080  
          resources:  
            limits:  
              cpu: "200m"  
              memory: "256Mi"  
            requests:  
              cpu: 100m  
              memory: 128Mi  
---  
apiVersion: v1  
kind: Service  
metadata:  
  annotations:  
  name: node-demo  
spec:  
  ports:  
    - name: node  
      port: 8080  
      targetPort: 8080  
      protocol: TCP  
  selector:  
    app: node  
  sessionAffinity: None  
  type: NodePort
```

```
mark@torre-ubuntu:~/Escritorio/kubernetes$ kubectl get all  
NAME                           READY   STATUS    RESTARTS   AGE  
pod/nginx-7b6999ccf4-8wzmp   1/1    Running   1 (1m ago)  76m  
pod/nginx-7b6999ccf4-lsdnf   1/1    Running   1 (1m ago)  76m  
pod/nginx-95544f78b-qkhsg   0/1    Completed  13 (5m7s ago)  76m  
pod/node-68cf7c49bb-sb8tw   1/1    Running   0          9m16s  
pod/node-68cf7c49bb-z8xzk   1/1    Running   0          9m16s  
  
NAME                  TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)           AGE  
service/kubernetes   ClusterIP   10.96.0.1   <none>        443/TCP          27d  
service/nginx-demo-service   NodePort   10.110.177.110  <none>        80:32062/TCP,81:31094/TCP  3d23h  
service/node-demo     NodePort   10.108.18.16   <none>        8080:31059/TCP       16m  
  
NAME                READY   UP-TO-DATE   AVAILABLE   AGE  
deployment.apps/nginx  2/2     1           2           33h  
deployment.apps/node   2/2     2           2           9m16s  
  
NAME                DESIRED  CURRENT   READY   AGE  
replicaset.apps/nginx-7b6999ccf4  2       2       2       33h  
replicaset.apps/nginx-95544f78b  1       1       0       31h  
replicaset.apps/node-68cf7c49bb  2       2       2       9m16s
```

Comprobamos que funciona



```
{"pedidos": [{"pollo": "1", "patatas": "1",
```

7. Instalación y configuración del entorno de desarrollo symphony (para la página de gestión de kubernetes y servicios)

Este servicio gestionará el estado de otros servicios y nos permitirá interacción con kubernetes se va a desarrollar en symfony por lo que necesitamos hacer de nuestro servidor un lamp

7.1. Instalación de php

Necesitaremos de php para poder utilizar este framework, evidentemente, por lo que procedemos a su instalación

```
sudo apt-get install apache2 php8.0 libapache2-mod-php8.0 -y
```

```
sudo apt-get install php8.0-common php8.0-mysql php8.0-xml php8.0-curl php8.0-odbc php8.0-imagine php8.0-xmlrpc php8.0-dev php8.0-imap php8.0-mbstring php8.0-opcache php8.0-soap php8.0-zip -y
```

Instalaremos también composer

7.2. Instalación de servidor web

Como para el anterior servicio web he utilizado nginx he pensado en utilizar apache para este

```
mark@torre-ubuntu:~/Escritorio/kubectl4everyone$ sudo apt install apache2
[sudo] contraseña para mark:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
```

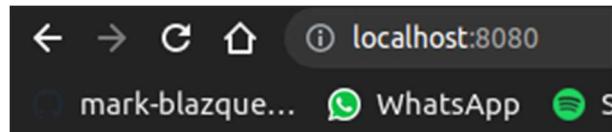
Empezare configurando el puerto de escucha de este que lo voy a cambiar

```
<VirtualHost *:8080>
    # The ServerName directive sets the request scheme, hostname and port
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file)
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    #
    # ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```

Reinicio el servicio y me aseguro que funciona



ejemplo

El siguiente paso es habilitar el funcionamiento de php en apache

```
sudo apt-get install php8.0-fpm libapache2-mod-fcgid -y
a2enmod proxy_fcgi setenvif
a2enconf php8.0-fpm
systemctl restart apache2
```

Y me aseguro de su funcionamiento haciendo el documento de php.info



7.3. Instalación mysql

Debemos de instalar un gestor de base de datos en mi caso será mysql

```
mark@torre-ubuntu:/var/www/html$ sudo apt install mysql-server
[sudo] contraseña para mark:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya

mark@torre-ubuntu:~/Escritorio$ sudo systemctl start mysql.service
[sudo] contraseña para mark:
mark@torre-ubuntu:~/Escritorio$ sudo mysql_secure_installation
```

Iniciamos el servicio y pasamos a la creación de un usuario para la gestión de la base de datos que también vamos a crear la cual almacenará usuario y contraseña

```
mysql> CREATE USER 'mark'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0,02 sec)
```

```
mysql> CREATE DATABASE usuarios;
Query OK, 1 row affected (0,01 sec)
```

Creamos la tabla

```
mysql> use usuarios
Database changed
mysql> create table credenciales (
    -> nombre VARCHAR(20) NOT NULL,
    -> password VARCHAR(20) NOT NULL,
    -> PRIMARY KEY ( nombre )
    -> );
Query OK, 0 rows affected (0,03 sec)
```

Le damos solo privilegios para insertar borrar y consultar al usuario creado para no trabajar con root

```
mysql> GRANT INSERT, DELETE, SELECT on *.* TO 'mark'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0,01 sec)
```

Una vez creado y con privilegios vamos a su sesión y realizaremos una inserción y consulta para ver si todo funciona como debe

```
mark@torre-ubuntu:~/Escritorio$ mysql -u mark -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 18
Server version: 8.0.29-0ubuntu0.21.10.2 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use usuarios
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> INSERT INTO credenciales (nombre,password) VALUES ("pepe","pepe")
-> ;
Query OK, 1 row affected (0,00 sec)
```

```
mysql> select * from credenciales
      -> ;
+-----+-----+
| nombre | password |
+-----+-----+
| pepe   | pepe     |
+-----+-----+
1 row in set (0,00 sec)
```

Se puede ver que todo funciona así que dejamos de momento esta parte

7.4. Instalación del framework

Por último, instalaremos symfony para trabajar con php

```
mark@torre-ubuntu:~$ wget https://get.symfony.com/cli/installer -O - | bash
--2022-05-07 20:18:52-- https://get.symfony.com/cli/installer
Resolving get.symfony.com (get.symfony.com)... 13.33.232.13, 13.33.232.111, 13.33.232.72, ...
Connecting to get.symfony.com (get.symfony.com)[13.33.232.13]:443... connected.
Peticion HTTP enviada, esperando respuesta... 200 OK
Longitud: 6161 (6,0K) [text/x-shellscrip]
Guardando como: 'STDOUT'

[  100%[=====]  2022-05-07 20:18:52 (173 KB/s) - escritos a stdout [6161/6161]

symfony CLI installer

Environment check
[*] cURL is installed
[*] Gzip is installed
[*] Git is installed
[*] Your architecture (amd64) is supported
```

Ya podemos iniciar el ejemplo de prueba y ver el resultado

```
mark@torre-ubuntu:~$ export PATH="$HOME/.symfony/bin:$PATH".
mark@torre-ubuntu:~$ source ~/.bashrc
mark@torre-ubuntu:~$ symfony new example --full
* Creating a new Symfony project with Composer
  WARNING: Unable to find Composer, downloading one. It is recommended to install it
  (running /home/mark/.symfony/composer/composer.phar create-project symfony)

* Setting up the project under Git version control
  (running git init /home/mark/example)

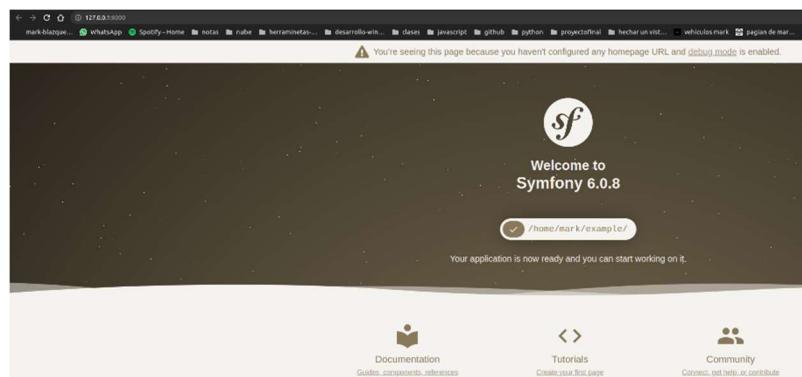
[OK] Your project is now ready in /home/mark/example

mark@torre-ubuntu:~$ cd example/
mark@torre-ubuntu:~/example$ symfony server start
No se ha encontrado la orden «symfony», pero se puede instalar con:
sudo apt install coinor-symphony
mark@torre-ubuntu:~/example$ symfony server:start

[WARNING] run "symfony server:ca:install" first if you want to run the web
          to avoid this warning

Tailing Web Server log file (/home/mark/.symfony/log/6ed151d2b144b39f42d9a0e
Tailing PHP log file (/home/mark/.symfony/log/6ed151d2b144b39f42d9a0e9fd47ea

[OK] Web server listening
    The Web server is using PHP CLI 8.0.8
    http://127.0.0.1:8000
```



Doy por concluido el proceso de instalación ya que vemos que todo funciona de forma correcta y pasamos al desarrollo de nuestra app

8. Desarrollo de la página de login para la página de gestión de servicios y kubernetes

Symfony dispone de un módulo que crea de forma rápida y muy sencilla un login magnifico, pero a la hora de consultar la opinión de programadores de utilizar procesos para obtener la información en vez de los métodos que ya vienen definidos comentaban que podía ser contraproducente así que mi solución va a ser la creación de una página php sin framework que realice esta verificación de usuario a través de un proceso

Comenzamos dando privilegios de creación de procedimientos al usuario

```
mysql> GRANT ALTER ROUTINE, CREATE ROUTINE, EXECUTE ON *.* TO 'mark'@'localhost' ;
Query OK, 0 rows affected (0,00 sec)
```

creo el procedimiento que va a seleccionar el nombre y contraseña de la base de datos

```
mysql> DELIMITER //
mysql> USE usuarios
Database changed
mysql> CREATE PROCEDURE seleccionar4 (
    -> IN nombre1 VARCHAR(20),
    -> contra1 VARCHAR(20)
    ->
    -> )
    -> BEGIN
    -> SELECT *
    -> FROM credenciales
    -> WHERE nombre = nombre1 and password = contra1;
    -> END//
Query OK, 0 rows affected (0,01 sec)
```

Compruebo que funciona

```
mysql> use usuarios
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CALL seleccionar4('pepe','pepe');
+-----+-----+
| nombre | password |
+-----+-----+
| pepe   | pepe     |
+-----+-----+
1 row in set (0,00 sec)

Query OK, 0 rows affected (0,00 sec)
```

con esto podemos pasar a php y aquí lo que haremos es capturar los datos que pasa el formulario de login para meterlos como parámetros en la consulta del procedure que básicamente no deja de ser otra cosa que una función

nombre

introduce nombre

contraseña

introduce contraseña

iniciar sesion

```
<?php
$nombre=$_POST["nombre"];
$contra=$_POST["contra"];

$dsn = 'mysql:usuarios;host=localhost:3306';
$usuario = 'mark';
$contrasena = "password";

try {
    $con = new PDO($dsn, $usuario, $contrasena);
    $con->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $sql = "CALL usuarios.seleccionar4(:nombre,:contra)";
    //publishers = [];

    $sentencia = $con->prepare($sql);
    $sentencia->bindValue(":nombre", $nombre);
    $sentencia->bindValue(":contra", $contra);
    // $sentencia->bindParam(':nombre', $nombre, PDO::PARAM_INT);
    $sentencia->execute();
    foreach( $sentencia as $row){
        echo $nombreseguro=$row[0];
        echo $contrasegura=$row[1];
    }
}
```

y el resultado es el esperado

pepepepe

En caso de ser erróneos las credenciales, no devuelve nada el procedure, así que hacemos la validación y con eso ya estaría hecho un login con procedure , el último paso será la redirección a la página de gestión de servicios cuando este definida y la definición de su entorno en apache, hospedamos la app de login en el puerto 8080

```

GNU nano 5.6.1                               /etc/apache2/sites-enabled/login.conf
<VirtualHost *:8080>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    #
    # ServerAdmin webmaster@localhost

    ##      DocumentRoot /var/www/html/kubectl4everyone/public
    DocumentRoot /var/www/html/login/
    ServerName loginpepe.com

        # It is also possible to configure the loglevel for particular
        # modules, e.g.

    # Use the front controller as index file. It serves as a fallback solution when
    # every other rewrite/redirect fails (e.g. in an aliased environment without
    # mod_rewrite). Additionally, this reduces the matching process for the
    # start page (path '/') because otherwise Apache will apply the rewriting rules
    # to each configured DirectoryIndex file (e.g. index.php, index.html, index.pl).
    #DirectoryIndex index.php

    # By default, Apache does not evaluate symbolic links if you did not enable this
    # feature in your server configuration. Uncomment the following line if you
    # install assets as symlinks or if you experience problems related to symlinks
    # when compiling LESS/Sass/Coffeescript assets.
    # Options +FollowSymlinks

    # Disabling Multiviews prevents unwanted negotiation, e.g. "/index" should not resolve
    # to the front controller "/index.php" but be rewritten to "/index.php/index".
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

```

9. Desarrollo de la página de gestión de servicios, kubernetes con symphony

Empezamos creando el proyecto

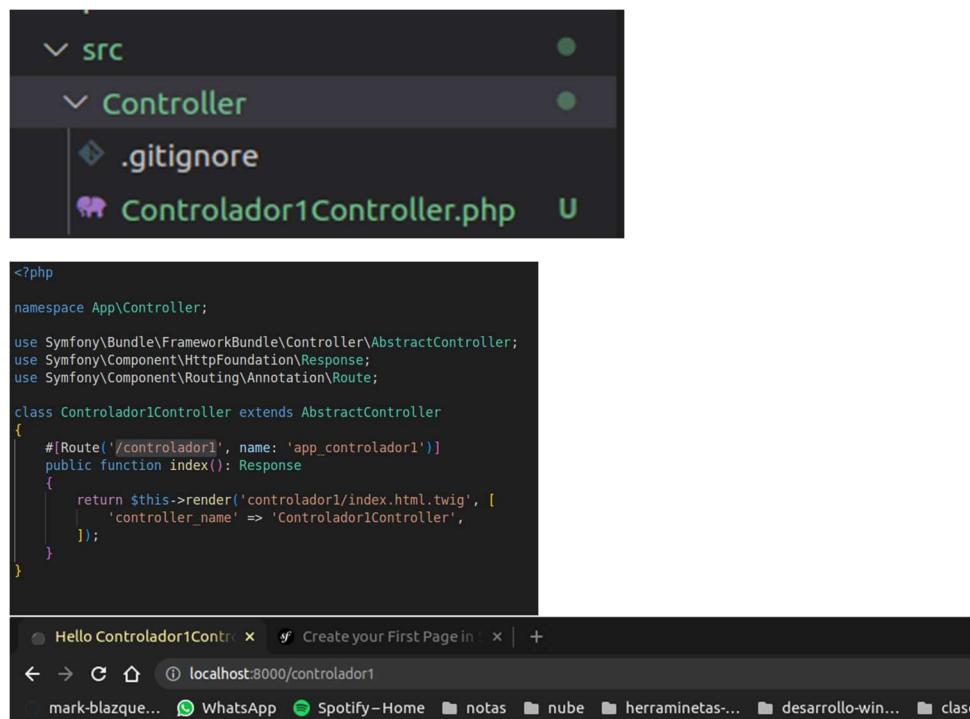
```
mark@torre-ubuntu:~/Escritorio$ symfony new kubectl4everyone --full
* Creating a new Symfony project with Composer
  WARNING: Unable to find Composer, downloading one. It is recommended to install
  Composer via https://getcomposer.org/installer
```

9.1. Creación de páginas

Funciona de forma muy parecida a larabel así que entendemos el uso del fichero routes y los controladores, así que empezaremos creando un controlador, pero antes exportando la ruta que si no provoca error

```
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ export PATH="$HOME/.symfony/bin:$PATH"
mark@torre-ubuntu:~/Escritorio/kubectl4everyone$ php bin/console make:controller controlador1
  created: src/Controller/Controlador1Controller.php
  created: templates/controlador1/index.html.twig
Success!
Next: Open your new controller class and add some pages!
```

Y podemos verlo en src



```
<?php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class Controlador1Controller extends AbstractController
{
    #[Route('/controlador1', name: 'app_controlador1')]
    public function index(): Response
    {
        return $this->render('controlador1/index.html.twig', [
            'controller_name' => 'Controlador1Controller',
        ]);
    }
}
```

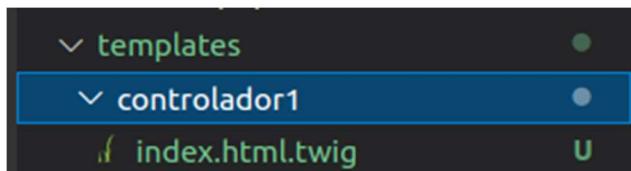
localhost:8000/controlador1

Hello Controlador1Controller! ✓

This friendly message is coming from:

- Your controller at [src/Controller/Controlador1Controller.php](#)
- Your template at [templates/controlador1/index.html.twig](#)

Aquí podemos ver la ruta de la vista que genera este controlador



Como he comentado mi deseo es obtener información de servicios tales como apache e información que los comandos de kubernetes nos proporciona así que eso es lo que voy a hacer

Cada controlador gestionara un comando linux para obtener esta información, se accederá a través de un index que tendrá botones que llevaran a una ruta específica que es donde se ejecutarán estos controladores.

Para el uso de comandos linux en symfony necesitamos de una clase llamada process así que procedemos a su instalación

```
mark@torre-ubuntu:~/Escritorio/kubectl4everyone$ composer require symfony/process
./composer.json has been updated
Running composer update symfony/process
Loading composer repositories with package information
Info from https://repo.packagist.org: #StandWithUkraine
```

Ya podemos ejecutar el ejemplo que nos proporciona la documentación oficial para ver el resultado, pero modificando para obtener información que nos interesa en este caso donde se ejecuta este comando

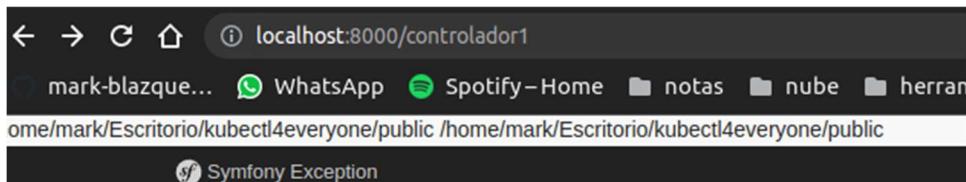
```
use Symfony\Component\Process\Exception\ProcessFailedException;
use Symfony\Component\Process\Process;

$process = new Process(['pwd']);
// $process = new Process(['ls -la']);

$process->run();

// executes after the command finishes
if (!$process->isSuccessful()) {
    throw new ProcessFailedException($process);
}

echo $process->getOutput();
```



Una vez comprobado que funciona pasamos a probar un comando que nos interese en este caso el de estado de kubernetes

```

$process = new Process(['/usr/bin/kubectl', 'get', 'all']);

$process->run();

// executes after the command finishes
if (!$process->isSuccessful()) {
    throw new ProcessFailedException($process);
}

echo $process->getOutput();

```

```

NAME READY STATUS RESTARTS AGE
pod/nginx-dd567d6bd-5vr6l 0/1 Pending 0 22h
pod/nginx-dd567d6bd-7mhzf 1/1 Terminating 2
(2d18h ago) 4d23h
pod/nginx-dd567d6bd-t6pvg 0/1 Pending 0 22h
pod/nginx2-5b4ccf66cd-5cztv 1/1 Terminating 1
(2d18h ago) 2d23h
pod/nginx2-5b4ccf66cd-hjz8w 0/1 Pending 0 22h
pod/nginx2-5b4ccf66cd-wwfws 1/1 Terminating 1
(2d18h ago) 2d22h
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/kubernetes ClusterIP 10.96.0.1 443/TCP 40d
service/nginx-demo-service NodePort 10.109.151.31 80:31317/TCP 5d22h
service/node-demo NodePort 10.108.18.16 8080:31
deployment.apps/nginx 0/2 2 0 5d23h
deployment.apps/nginx2 0/2 2 0 5d22h
deployment.apps/node 0/1 1 0 12d
NAME DESIRED CUR
replicaset.apps/nginx2-5b4ccf66cd 2 2 0 2d23h
replicaset.apps/nginx2-8469bf47c5 0 0 0 5d22h
replicaset.apps/node-64cc6c867 0 0 0 1

```

Se observa que el comando se ejecuta de forma correcta, por lo que llegamos a la conclusión de que de esta forma vamos a poder gestionar ciertos servicios de la forma deseada así que ahora solo debemos de dar un formato legible ya que parece ser que mete todo en un array y lo imprime

Así que lo que hago es meter el proceso dentro de la creación de la vista del controlador y declaro como variable el resultado

```

class Controlador1Controller extends AbstractController
{
    #[Route('/controlador1', name: 'app_controlador1')]
    public function index(): Response
    {
        $process = new Process(['/usr/bin/kubectl', 'get', 'all']);

        $process->run();

        // executes after the command finishes
        if (!$process->isSuccessful()) {
            throw new ProcessFailedException($process);
        }

        $kubernetes = $process->getOutput();
    }
}

```

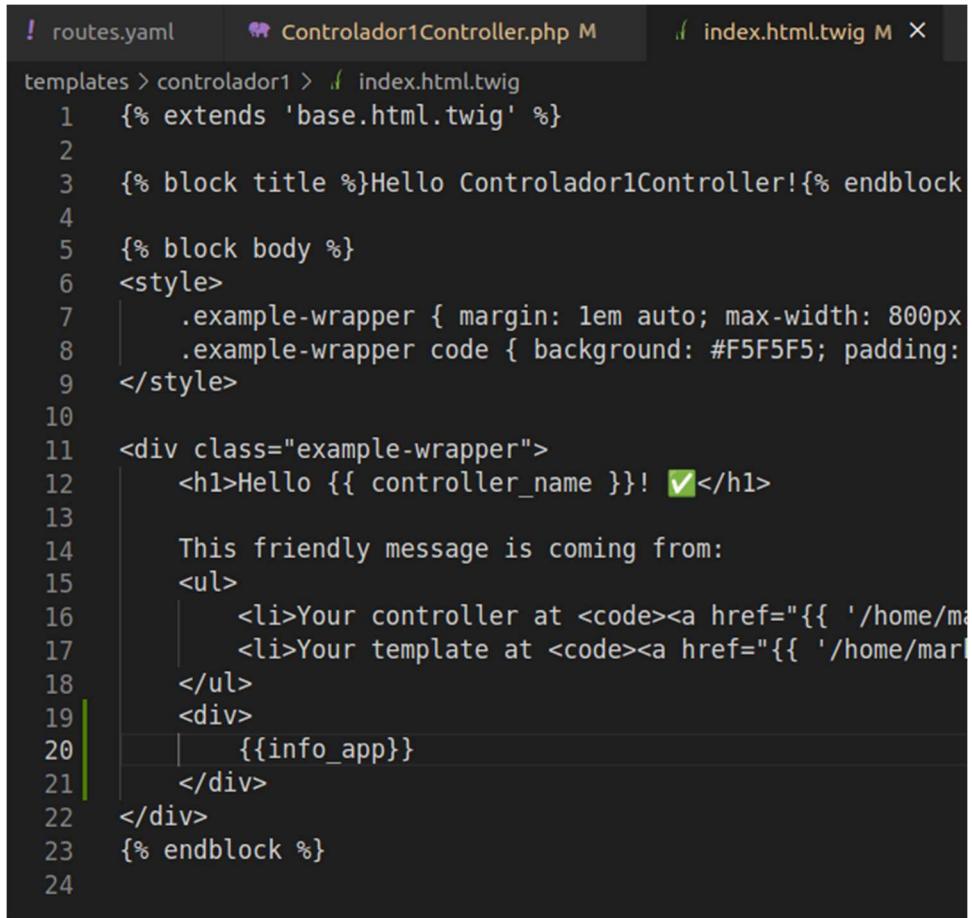
Una vez hecho esto como symfony en su versión actual trabaja con twig para renderizar el html lo que hago es declarar info_app como el valor de la variable y luego esta se pasa al front

```

return $this->render('controlador1/index.html.twig', [
    'controller_name' => 'Controlador1Controller',
    'info_app' => $kubernetes ,
]);

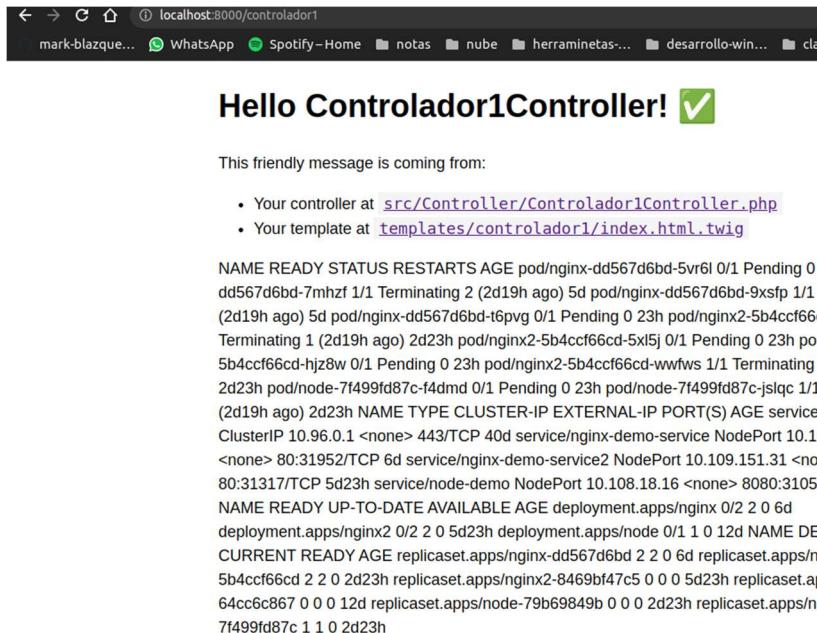
```

Y en el front lo imprimimos



```
routes.yaml          Controlador1Controller.php M          index.html.twig M X
templates > controlador1 > index.html.twig
1  {% extends 'base.html.twig' %}
2
3  {% block title %}Hello Controlador1Controller!{% endblock %}
4
5  {% block body %}
6      <style>
7          .example-wrapper { margin: 1em auto; max-width: 800px }
8          .example-wrapper code { background: #F5F5F5; padding:
9      </style>
10
11     <div class="example-wrapper">
12         <h1>Hello {{ controller_name }}! ✓</h1>
13
14         This friendly message is coming from:
15         <ul>
16             <li>Your controller at <code><a href="{{ '/home/mark-blazquez/controle...'}}</a></code>
17             <li>Your template at <code><a href="{{ '/home/mark-blazquez/controle...'}}</a></code>
18         </ul>
19         <div>
20             {{info_app}}
21         </div>
22     </div>
23     {% endblock %}
```

El resultado



Solo nos falta darle un diseño legible y para eso lo que hacemos es pasarla por la función que añade salto de líneas entre estos valores que se aplicara en el php eol que significa el end of lines

```
$kubernetes = str_replace(PHP_EOL, '<br>', $kubernetes);
```

Y en el front

```
<code>{{ info_app | raw }}</code>
```

Y así nos quedaría el resultado

Hello Controlador1Controller! ✓

This friendly message is coming from:

- Your controller at [src/Controller/Controlador1Controller.php](#)
- Your template at [templates/controlador1/index.html.twig](#)

```
NAME READY STATUS RESTARTS AGE
pod/nginx-dd567d6bd-5vr6l 0/1 Pending 0 24h
pod/nginx-dd567d6bd-7mhzf 1/1 Terminating 2 (2d20h ago) 5d1h
pod/nginx-dd567d6bd-9xsf 1/1 Terminating 2 (2d20h ago) 5d1h
pod/nginx-dd567d6bd-t6pv 0/1 Pending 0 24h
pod/nginx2-5b4ccf66cd-5cztv 1/1 Terminating 1 (2d20h ago) 3d
pod/nginx2-5b4ccf66cd-5xl5j 0/1 Pending 0 24h
pod/nginx2-5b4ccf66cd-hjz8w 0/1 Pending 0 24h
```

Mucho más legible y entendible pero no aun del todo perfecto ya que no casa con las casillas así que decido ver lo que pasa y resulta que trae de la terminal los espacios en blancos ya que lo muestra como un string y si añadimos una pieza más de código lo veremos más claro

```
$kubernetes = str_replace(" ", "_", $kubernetes);
```

```
NAME READY STATUS RESTARTS AGE
nginx-dd567d6bd-5vr6l 0/1 Pending 0 26h
nginx-dd567d6bd-7mhzf 1/1 Terminating 2 (2d22h ago) 5d3h
nginx-dd567d6bd-9xsf 1/1 Terminating 2 (2d22h ago) 5d3h
nginx-dd567d6bd-t6pv 0/1 Pending 0 26h
nginx2-5b4ccf66cd-5cztv 1/1 Terminating 1 (2d22h ago) 3d2h
nginx2-5b4ccf66cd-5xl5j 0/1 Pending 0 26h
nginx2-5b4ccf66cd-hjz8w 0/1 Pending 0 26h
nginx2-5b4ccf66cd-wwfws 1/1 Terminating 1 (2d22h ago) 3d2h
node-7f499fd87c-f4dmd 0/1 Pending 0 26h
node-7f499fd87c-jslqc 1/1 Terminating 1 (2d22h ago) 3d2h
```

Evidente mente esto es horrible así que intento sustituir todo esto y meterlo dentro de una tabla para que quede legible y bonito y para ello lo que hago es sustituir todas las barras bajas por cierre e inicios de tablas html evidentemente tengo que contemplar toda la cantidad de estas ya que vienen dadas por terminal, veamos el código

```
$kubernetes = str_replace("____", "</td><td>", $kubernetes);
//$kubernetes = str_replace("____", "</td><td>", $kubernetes);
$kubernetes = str_replace('<br>', '<tr><td>', $kubernetes);
```

Y el front end quedaría así

```
<div>
  <table >
    <tr>
      |   <td>{{ info_app | raw }}</td>
    </tr>
  </table>
</div>
```

Y en el último lo que hago es sustituir los saltos de líneas por comienzo de nueva fila de tabla y abrir nueva celda dando un bonito y ordenado resultado

NAME	READY STATUS	RESTARTS	AGE
nginx-dd567d6bd-5vr6l	0/1	Pending	0 26h
nginx-dd567d6bd-7mhzf	1/1	Terminating	2_(2d22h_ago) 5d4h
nginx-dd567d6bd-9xsfp	1/1	Terminating	2_(2d22h_ago) 5d4h
nginx-dd567d6bd-t6pvg	0/1	Pending	0 26h
nginx2-5b4ccf66cd-5cztv	1/1	Terminating	1_(2d22h_ago) 3d3h
nginx2-5b4ccf66cd-5xl5j	0/1	Pending	0 26h
nginx2-5b4ccf66cd-hjz8w	0/1	Pending	0 26h
nginx2-5b4ccf66cd-wwfws	1/1	Terminating	1_(2d22h_ago) 3d3h
node-7f499fd87c-f4dmd	0/1	Pending	0 26h
node-7f499fd87c-jslqc	1/1	Terminating	1_(2d22h_ago) 3d2h

Ahora si finalizado el resultado de forma satisfactoria comienzo a realizar todos los demás controladores que realizaran actividades parecidas

He metido bootstrap con la instalación del módulo correspondiente que me servirá para darle cierto formato de forma rápida no obstante añadiré algo de código CSS para diferenciar el estilo de la página y dotarla de personalidad

Por lo que la aplicación quedaría dividida de la siguiente forma

Controlador 1 –get pod

Controlador 2 – get service

Controlador3- get deployment

Controlador4-get replicaset

Controlador 5 – nginx

Al hacer los 4 primeros controladores casi la misma función me centrare en mostrar el 5 el de nginx ya que difiere del resto

9.2. Página de nginx

En nginx la cosa cambia, ya que el process hace un systemctl status mostrando su estado y pasa al index los datos de forma diferente sin problemas de los espacios como antes pasaba

```
$process = new Process(['/usr/sbin/service', 'nginx', 'status']);

$process->run();

// executes after the command finishes
if (!$process->isSuccessful()) {
    throw new ProcessFailedException($process);
}

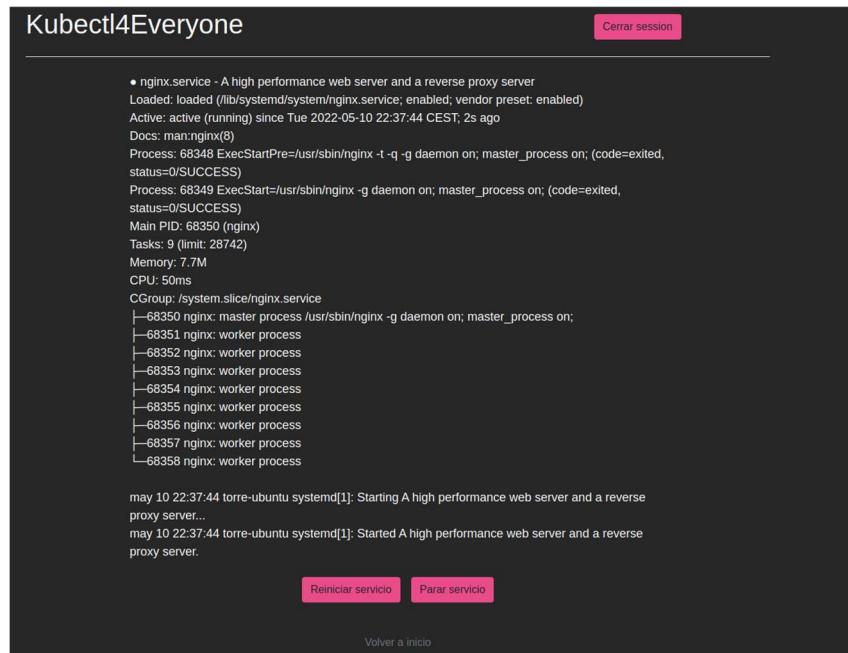
$nginx = $process->getOutput();
$nginx = str_replace(PHP_EOL, '<br>', $nginx);

return $this->render('controlador5/index.html.twig', [
    'controller_name' => 'Controlador1Controller',
    'info_app' => $nginx ,
    1);
```

Pasa la info al index y este contiene otros dos botones que nos permite el reinicio del servicio y la parada que no será más que otros dos controladores con esta función

```
<div class="example-wrapper">
    <h1 class="text-light text-center">Nginx</h1>

    <div>
        <p class="text-light">{{ info_app | raw }}</p>
    </div>
    <div class="d-flex justify-content-center">
        <div class="p-2">
            <a href="/reiniciar" ><button id="boton" class="btn" role="button">Reiniciar servicio</button></a>
        </div>
        <div class="p-2">
            <a href="/parar" ><button id="boton" class="btn" role="button">Parar servicio</button></a>
        </div>
    </div>
</div>
```



Veamos los controladores de reinicio y parada

```
^CShut down, bye!
mark@torre-ubuntu:~/Escritorio/kubectl4everyone$ php bin/console make:controller reiniciar
  created: src/Controller/ReiniciarController.php
  created: templates/reiniciar/index.html.twig
```

Success!

```
Next: Open your new controller class and add some pages!
mark@torre-ubuntu:~/Escritorio/kubectl4everyone$ php bin/console make:controller parar
```

```
$process = new Process(['/usr/sbin/service', 'nginx', 'stop']);
$process->run();

if (!$process->isSuccessful()) {
    throw new ProcessFailedException($process);
}

return $this->render('parar/index.html.twig', [
    'mensaje' => 'accion realizada, parada de servicio correcta',
]);
```

Y el index mostraría el mensaje con el botón para ir de nuevo a la gestión del servicio

```
<div class="example-wrapper">
  <div class="alert alert-warning text-center" role="alert">
    {{ mensaje }}<br>
    <a class="text-muted" href="/nginx" ><p>volver a la pagina de Nginx</p></a>
  </div>
</div>
```

Cuando realizo la acción de reinicio o parada mi servidor me vuelca la ventana para meter la clave de sudo ,como es normal ya que esta acción las requiere , así que veo con que usuario hace esta tarea modificando el process

```
$process = new Process(['whoami']);

$process->run();

// executes after the command finishes
if (!$process->isSuccessful()) {
    throw new ProcessFailedException($process);
```



Veo que vuelca mi usuario así que lo que primero se me ocurre es modificar el sudoers para que mi usuario pueda realizar estas acciones sin problema y con permisos de sudo sin contraseña para que la acción no se detenga hasta la introducción de la contraseña que es lo que está pasando ahora.

Después de unas pruebas veo que el resultado es satisfactorio, pero como voy a pasar la app al servidor apache voy a detenerme en este punto y mostrarlo con el usuario de apache www-data

9.3. Paso de aplicación a producción en apache

Antes lo veíamos todo bien por el hecho de que todo iba en el servidor que trae por defecto symfony y trabajaba en el escritorio, al hacer algunos cambios de permisos y usuarios lo he roto así que primero borramos estos directorios

```
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ sudo rm -fr var/log/*
[sudo] contraseña para mark:
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ sudo rm -fr var/cache/*
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ sudo setfacl -dR -m u:mark:rwx -m u:'whoami':rwx var/cache var/log
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ sudo setfacl -R -m u:www-data:rwx -m u:'whoami':rwx var/cache var/log
```

El error es causado porque necesita de permisos de escritura para los logs y cache ubicado en /var dentro del proyecto y hay conflicto con el usuario del servidor web y el que ejecuta la app todo esto es causado a que lo he movido al directorio de apache para preparar las primeras pruebas para ver si todo funciona bien fuera del servidor symfony

Dejamos tal que así el directorio de apache con la app de login y la de gestión de servicios separadas

```
mark@torre-ubuntu:/var/www/html$ ls
index.html  info.php  kubectl4everyone  login
mark@torre-ubuntu:/var/www/html$
```

Creamos otro archivo de configuración para la app d servicios o sea la de symfony y lo modificamos preparando el entorno para poner la aplicación en producción

```
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ composer require symfony/apache-pack
Using version ^1.0 for symfony/apache-pack
./composer.json has been updated
Running composer update symfony/apache-pack
Loading composer repositories with package information
Updating dependencies
```

Este comando crea unas reglas en el directorio public en el fichero.htacces y estas han de ser copiadas en el archivo de configuración de apache

```
GNU nano 5.6.1
VirtualHost *:8085
DocumentRoot /var/www/html/kubectl4everyone/public
<Directory /var/www/html/kubectl4everyone/public>
    AllowOverride None
    # Copy .htaccess contents here
    # Use the front controller as index file. It serves as a fallback solution when
    # every other rewrite/redirect fails (e.g. in an aliased environment without
    # mod_rewrite). Additionally, this reduces the matching process for the
    # start page (path "/") because otherwise Apache will apply the rewriting rules
    # to each configured DirectoryIndex file (e.g. index.php, index.html, index.pl).
    DirectoryIndex index.php

    # By default, Apache does not evaluate symbolic links if you did not enable this
    # feature in your server configuration. Uncomment the following line if you
    # install assets as symlinks or if you experience problems related to symlinks
    # when compiling LESS/Sass/CoffeeScript assets.
    # Options +FollowSymlinks

    # Disabling MultiViews prevents unwanted negotiation, e.g. "/index" should not resolve
    # to the front controller "/index.php" but be rewritten to "/index.php/index".
    <IfModule mod_negotiation.c>
        Options -MultiViews
    </IfModule>

    <IfModule mod_rewrite.c>
        RewriteEngine On
        # Determine the RewriteBase automatically and set it as environment variable.
        # If you are using Apache aliases to do mass virtual hosting or installed the
        # project in a subdirectory, the base path will be prepended to allow proper
        # resolution of the index.php file and to redirect to the correct URI. It will
        # work in environments without path prefix as well, providing a safe, one-size
        # fits all solution. But as you do not need it in this case, you can comment
        # the following 2 lines to eliminate the overhead.
        RewriteCond %{REQUEST_URI}::$1^/(.+)/(.*):$2
        RewriteRule .* - [E=BASE:$1]
        # Sets the HTTP_AUTHORIZATION header removed by Apache
        RewriteCond %{HTTP:Authorization} .
        RewriteRule ^ - [E=HTTP_AUTHORIZATION:$0]

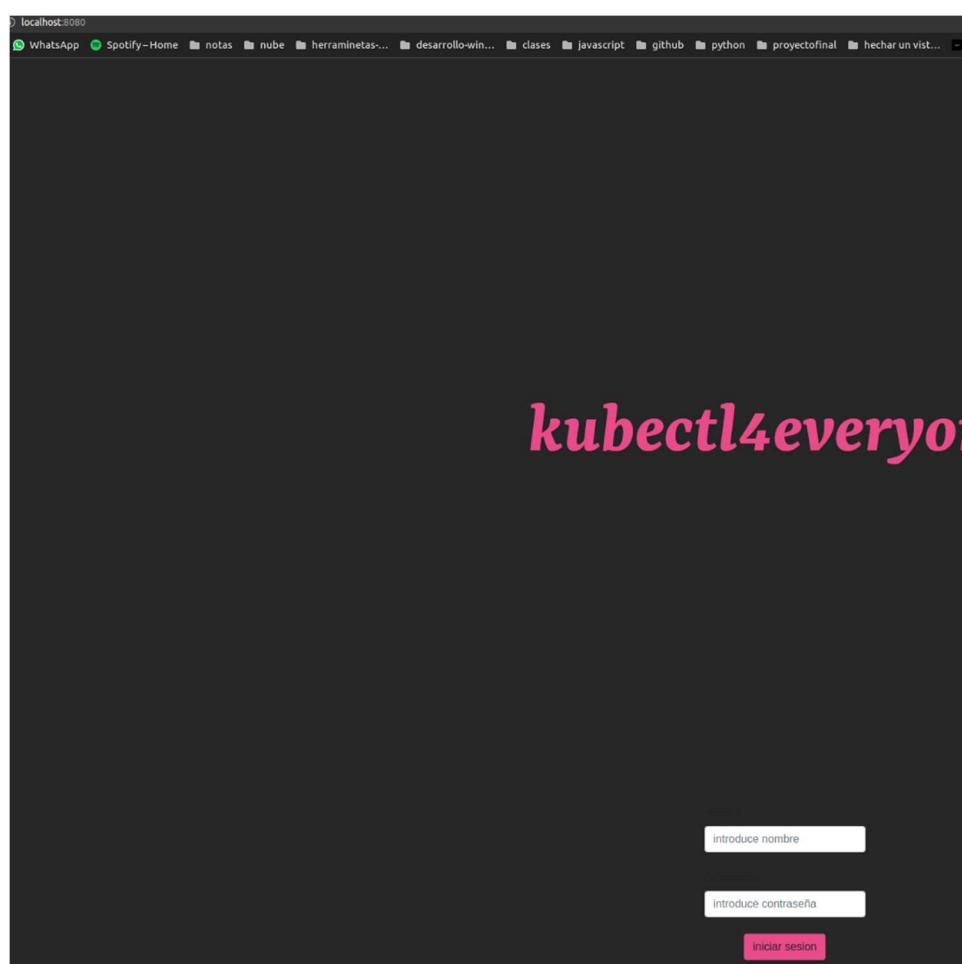
        # Redirect to URI without front controller to prevent duplicate content
        #  
        RewriteCond %{ENV:REDIRECT_STATUS} =200
        #  
        RewriteRule ^(.*)$ /$1 [L,R=301]
```

Nos queda habilitar los sitios

```
mark@torre-ubuntu:~$ sudo a2ensite login.conf
Site login already enabled
mark@torre-ubuntu:~$ sudo a2ensite servicios.conf
Site servicios already enabled
```

Y ya podemos poner la dirección de en la app de login cuando es correcto para que redireccione de una app a otra y en caso de que la consulta al procedure venga vacía nos lleve a la pag de error

```
if ($nombreseguro==""){
    header("location:pagerror.html");
}else{
    if($nombre==$nombreseguro){
        if($nombre==$nombreseguro){
            header("location:http://localhost:8081");
        }
    }
}
```



Y nos redirige a la página debida, el index, en caso correcto sino a la de error

[POD](#)[SERVICE](#)[DEPLOYMENT](#)[REPLICASET](#)[NGINX](#)

Lo sentimos pero no te has logueado de forma correcta vuelve a intentarlo

[volver a inicio](#)

Pero cuando voy a ejecutar una función como por ejemplo mostrar los pod me proporciona un error

Error Output:
=====

```
error: the server doesn't have a resource type "pod"
```

Exception Logs 1 Stack Trace

Symfony\Component\Process\Exception
ProcessFailedException

in src\Controller\Controller1Controller.php (line 23)

```
18.     $process->run();
19.
20.
21.     // executes after the command finishes
22.     if (!$process->isSuccessful()) {
23.         throw new ProcessFailedException($process);
24.     }
25.
26.     $kubernetes = $process->getOutput();
27.     $kubernetes = str_replace(PHP_EOL, '<br>', $kubernetes);
28.     $kubernetes = str_replace(' ','_',$kubernetes);
```

in vendor/symfony/http-kernel/HttpKernel.php -> **index** (line 152)
in vendor/symfony/http-kernel/HttpKernel.php -> **handleRaw** (line 74)
in vendor/symfony/http-kernel/Kernel.php -> **handle** (line 202)
in vendor/symfony/runtime/Runner/Symfony/HttpKernelRunner.php -> **handle** (line 35)
in vendor/autoload_runtime.php -> **run** (line 29)

Sospecho que es problema para ejecutar el process así que hago que el comando me muestre que usuario está ejecutándolo



Y efectivamente al estar hospedado en apache ha pasado a ser propiedad de www-data por lo que debo de dar permisos a este usuario para que ejecute estas funcionalidades de kubernetes con el comando kubectl

```
mark@torre-ubuntu:~$ sudo mkdir /var/www/.kube
mark@torre-ubuntu:~$ sudo cp -i /etc/kubernetes/admin.conf /var/www/.kube/config
mark@torre-ubuntu:~$ chown www-data:www-data /var/www/.kube/config
chown: cambiando el propietario de '/var/www/.kube/config': Operación no permitida
mark@torre-ubuntu:~$ sudo chown www-data:www-data /var/www/.kube/config
```

Y efectivamente ya nos permite ejecutar este comando

Kubectl4Everyone

NAME

nginx-dd567d6bd-5vr6l

nginx-dd567d6bd-m6v86

nginx-dd567d6bd-qtwtj

Para el botón de reiniciar nginx, necesito los permisos de sudo y para ello meto esto en sudoers como mencionaba con anterioridad

```
root    ALL=(ALL:ALL) ALL
www-data ALL=(ALL) NOPASSWD:ALL
www-data torre-ubuntu = (root) NOPASSWD: /usr/sbin/service nginx stop
ALL ALL=NOPASSWD: /usr/sbin/service nginx stop
```

Y en la app modificamos poniendo el comando sudo

```
$process = new Process(['sudo', '/usr/sbin/service', 'nginx', 'restart']);
$process->run();
```

Y ya me permite realizar la acción de forma correcta tanto el reinicio como la parada

9.4. Cookie para mantener sesiones

Vamos a mejorar el inicio de sesión para que solo puedas acceder a esta página si existe una cookie

En la verificación del usuario creamos la cookie

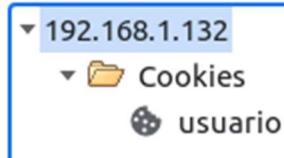
```
setcookie("usuario",time()+(365*24*60*60));
```

Y este es el resultado

Cookies en uso

Permitido

Se han permitido las siguientes cookies al \



Y para comprobar si existe la cookie deberemos de poner esta sección de código en todos los controladores

```
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\HttpFoundation\Request;

use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\HttpFoundation\RedirectResponse;

class IndexController extends AbstractController
{
    #[Route('/index', name: 'app_index')]
    public function index(Request $request)
    {
        $cookies = $request->cookies;

        if ($cookies->has('usuario')){
            return $this->render('index/index.html.twig', [
                'controller_name' => 'IndexController',
            ]);
        }else {
            return new RedirectResponse('https://192.168.1.132:8080');
        }
    }
}
```

Donde el return será diferente para cada uno que lo único que hace es devolver la vista pedida y algún tipo de información en el caso de los process y en caso de no existir lo que nos dejaría claro que no ha habido un inicio de sesión por parte del usuario hace una redirección a la página de login

También vamos a añadir en el inicio de sesión en el index esta sección de código para si en caso de que exista la cookie no haga falta iniciar sesión de nuevo

```
<?php
if(isset($_COOKIE["usuario"])){
    header("location:https://192.168.1.132:8085");
}
?>
<!DOCTYPE html>
```

9.5. Cierre de sesión

Para el cierre de sesión lo que haremos es destruir esa cookie con el método correspondiente

Para ello apuntamos con el botón a la ruta correspondiente

```
<a href="/cerrar" ><button id="boton" class="btn" role="button">Cerrar session</button></a>
```

Creamos el controlador y le asociamos su ruta

```
mark@torre-ubuntu:~/var/www/html/kubectl4everyone$ php bin/console make:controller cerrar
  created: src/Controller/CerrarController.php
  created: templates/cerrar/index.html.twig
```

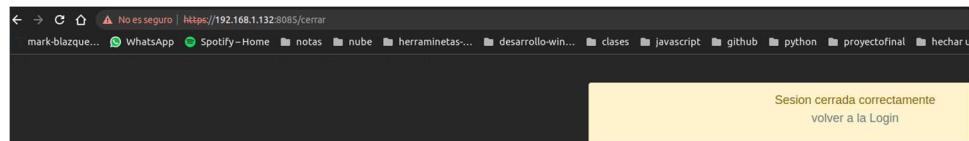
Este controlador será sencillo ya que lo único que hará será redireccionar a la página de login y lo más importante destruir la cookie antes

```
$view = $this->render('cerrar/index.html.twig', [
    'controller_name' => 'CerrarController',
]);

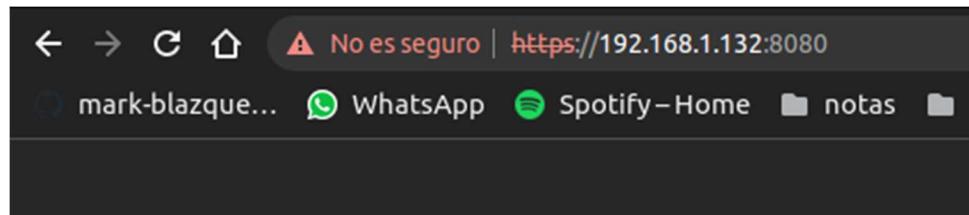
$res = new Response($view);
$res->headers->clearCookie('usuario');
$res->send();
```

La traducción del código es que el servidor responde con una vista y con una cabecera que borra la cookie

Lo que hace es devolver una vista que tiene un enlace para volver al login y en ese mismo instante borra la cookie



Y nos devuelve de forma correcta



Recordemos que si la cookie no se borra no podemos acceder a esta página de ninguna manera ya que esta comprueba si existe y nos dirige a la página de gestión

9.6. Funcionalidad extra: cambio de numero de pod

Como ya he comentado el diseño de esta aplicación es el evitar el uso de ssh lo máximo posible así que para ello voy a implementar la función de kubernetes de aumento o disminución de pods a través de un sencillo formulario

```
<div class="example-wrapper">
  <h1 class="text-light">Formulario de cambio de numero de pod</h1>

  <form action=". /modificar" method="post">
    <div class="form-group">
      <select class="form-select" name="nombre" placeholder="introduce nombre del deployment" required>
        <option value="nginx">nginx-pagina con index</option>
        <option value="nginx2">nginx2-pagina de gestion de tienda</option>
      </select>
      <input type="number" class="form-control" name="numero" placeholder="introduce numero de pod" required><br>
      <div class="d-flex justify-content-center">
        <input type="submit" id="boton" class="btn" value="modificar">
      </div>
    </div>
  </form>
</div>
```

Formulario de cambio de numero de pod

nginx-pagina con index

introduce numero de pod

modificar

Y creamos el controlador donde va a apuntar este formulario

```
mark@torre-ubuntu:/var/www/html/kubectl4everyone$ php bin/console make:controller modificar
  created: src/Controller/ModificarController.php
  created: templates/modificar/index.html.twig

Success!

Next: Open your new controller class and add some pages!
```

La forma de recuperar los datos de post en symfony es así

```
$request2 = Request::createFromGlobals();

$cookies = $request->cookies;
$nombre=$request2->request->get('nombre');
$numero=$request2->request->get('numero');

echo $nombre;
echo $numero;
```

Y podemos ver los datos

nginx1

Con eso hecho ya podemos hacer el proceso que modifique el número de pod con los datos que le hemos pasado

```

if ($cookies->has('usuario')) {
    $request2 = Request::createFromGlobals();

    $nombre=$request2->request->get('nombre');
    $numero=$request2->request->get('numero');

    //echo $nombre;
    //echo $numero;
    $process = new Process(['/usr/bin/kubectl','scale',"deployment.apps/$nombre","--replicas=$numero"]);

    $process->run();

    if (!$process->isSuccessful()) {
        throw new ProcessFailedException($process);
    }

    return $this->render('modificar/index.html.twig', [
        'mensaje' => 'accion realizada, numero de pod modificado',
    ]);
} else {
    return new RedirectResponse('https://torre-ubuntu.ddns.net/:8080');
}

```

Se puede ver como el comando se debe de meter tal y como kubernetes lo entiende y se le pasan las variables, mucha atención a cuando se utilizan comillas dobles o simples, es evidente que alguien se puede equivocar a la hora de escribir el nombre es por ello que para insertar el deployment que se quiere va por menú de selección

Y como vemos pasa el nombre en el value del formulario

```

<select class="form-select" name="nombre" placeholder="Introduce el nombre del pod que deseas modificar" style="width: 100%; height: 30px; border-radius: 5px; border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;">
    <option value="nginx">nginx-pagina con index</option>
    <option value="nginx2">nginx2-pagina de gestion de tienda</option>

```

Compruebo funcionamiento

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
nginx	2/2	2	2	23d
nginx2	2/2	2	2	23d
node	1/1	1	1	30d

 Below this is another form titled 'Formulario de cambio de numero de pod' with a dropdown menu showing 'nginx-pagina con index' and a pink 'modificar' button."/>

accion realizada, numero de pod modificado
[volver a la pagina de Deployment](#)

Y efectivamente lo realiza

Deployment

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
nginx	1/1	1	1	23d
nginx2	2/2	2	2	23d
node	1/1	1	1	30d

Con esto se concluye el desarrollo de esta aplicación ya que el siguiente paso es mostrar cómo se incorporan los certificados (que para este punto ya lo había hecho, pero me gusta seccionar y explicar cada cosa en su sitio) y el dominio

10. Dns público: no ip

Añadiremos la resolución por nombre para nuestro servidor, pero me encuentro con el problema de que mi compañía(isp) no me proporciona una ip publica para ipv4 debido a que hacen dnat de forma predeterminada y tampoco ofrecen una dirección ipv6, pero la solución ha sido fácil una llamadita y solicitarla, como dato curioso contaré que el trabajador que me atendió estaba con la firme convicción de llamar a la ip publica privada, me puso muy nervioso, esto no pasaría si hubiese estudiado.

10.1. Adopción del servicio

He seleccionado para cubrir mi necesidad el servicio de noip para implantarlo iremos a la web y configuraremos con nuestros datos



Descargamos el programa que se nos indica y descomprimimos en el directorio /usr/local/bin

```
mark@torre-ubuntu:/usr/local/bin$ sudo rm noip-duc-linux.tar.gz
```

Y haremos el comando make install

```
if [ ! -d /usr/local/bin ]; then mkdir -p /usr/local/bin;fi
if [ ! -d /usr/local/etc ]; then mkdir -p /usr/local/etc;fi
cp noip2 /usr/local/bin/noip2
/usr/local/bin/noip2 -C -c /tmp/no-ip2.conf

Auto configuration for Linux client of no-ip.com.

Multiple network devices have been detected.

Please select the Internet interface from this list.

By typing the number associated with it.
0      enp0s31f6
1      docker0
2      flannel.1
3      cni0
0

Please enter the login/email string for no-ip.com blazquezmark97@gmail.com
Please enter the password for user 'blazquezmark97@gmail.com' ****

Only one host [torre-ubuntu.ddns.net] is registered to this account.
It will be used.
Please enter an update interval:[30]
Do you wish to run something at successful update?[N] (y/N) n

New configuration file '/tmp/no-ip2.conf' created.

mv /tmp/no-ip2.conf /usr/local/etc/no-ip2.conf
mark@torre-ubuntu:/usr/local/bin/noip-2.1.9-1$
```

E iniciamos el servicio en segundo plano

```
mark@torre-ubuntu:/usr/local/bin/noip-2.1.9-1$ sudo /usr/local/bin/noip2
One noip2 process is already active,
and the multiple instance flag (-M) is not set.
```

Toca hacer el portforwarding en el router para comprobar si tenemos conectividad a nuestra ip publica para luego utilizar nuestro dns, evidentemente habilitare la dmz a mi servidor principal

Utilizare el puerto 86 que es el que tengo para una página de nginx

Using Hostname
torre-ubuntu.ddns.net

to connect to
5655

Port Forwarding

Hostname:
torre-ubuntu.ddns.net

Testing IP: 213.194.165.62

Port Forwarding

Enter a port to check:
86

Port 86 is open

Need help? Go Back Next Step

Con esto ya podemos ver que la configuración de puerto funciona y ahora nos aseguraremos del nombre

```
mark@torre-ubuntu:/usr/local/bin/noip-2.1.9-1$ nslookup torre-ubuntu.ddns.net
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:  torre-ubuntu.ddns.net
Address: 213.194.165.62
```

Una vez comprobado que todo funciona es hora de hacer la configuración para mis páginas, recordemos que hay que hospedar cuatro aplicaciones, 3 de kubernetes y la de gestión de servicios

10.2. Uso en la app de apache

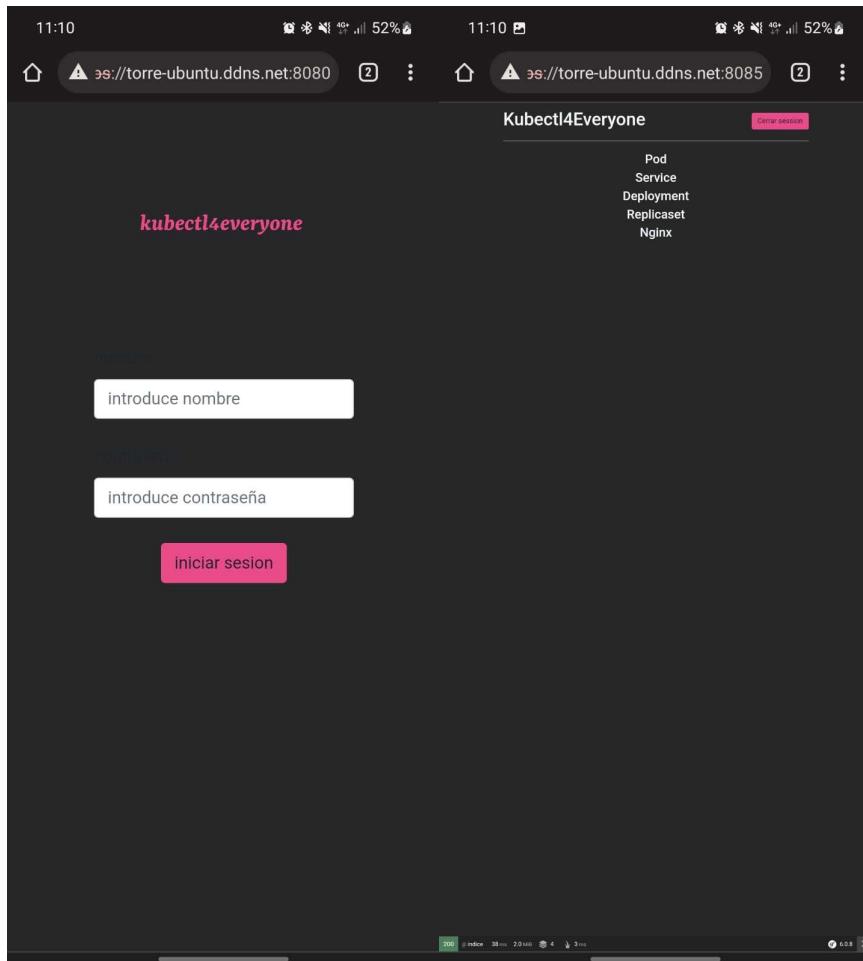
Mi idea es clara, en el puerto 8080 que hospedará la de gestión de servicios, que coincidirá con 8080 de mi servidor y el 8085

ON	servicio	TCP	*	192.168.1.132	8080	8080		
ON	gestionservicios	TCP	*	192.168.1.132	8085	8085		

Con esto ya podemos realizar los cambios en mi aplicación para probar que la aplicación de apache se ofrece de forma adecuada a través de nombre, así que sustituiremos todas las ip que había por el nombre de dominio de mi publica como ejemplo pondré la página de inicio de sesión que hace redirección a otro puerto y me parece más interesante

```
if ($contraseña = $contrasegura) {  
    header("location:https://torre-ubuntu.ddns.net:8085");  
    //crear cookie para la verificación de la pag de symfony  
    setcookie("usuario", time() + (365 * 24 * 60 * 60));  
}
```

Y haremos la comprobación con la conexión de datos de mi teléfono para asegurarme de que todo va bien



Y podemos ver que funciona perfectamente

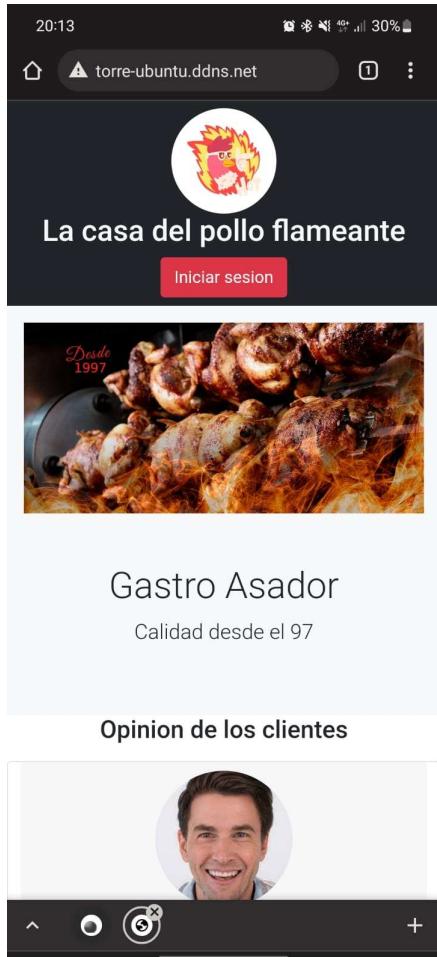
10.3. Uso en las apps de kubernetes

Para las de kubernetes será tan simple como hacer el port forward de nuestro router (dominio) a nuestro servicio node port (que es el que yo voy a utilizar por diversas razones, entre una de ellas la simplicidad que nos ofrece)

<input checked="" type="checkbox"/> ON	pag asador index	TCP	*	192.168.1.132	80	31952		
<input type="checkbox"/> OFF	pag asador index ssl	TCP	*	192.168.1.132	443	31952		
<input type="checkbox"/> OFF	paggestionasadorssl	TCP	*	192.168.1.132	444	31317		
<input checked="" type="checkbox"/> ON	paggestionasador	TCP	*	192.168.1.132	81	31317		
<input checked="" type="checkbox"/> ON	gestionservicios	TCP	*	192.168.1.132	8085	8085		
<input checked="" type="checkbox"/> ON	node	TCP	*	192.168.1.132	31059	31059		

La elección de los puertos es simple y es si hacen una petición a mi dominio quiero que lo que se ofrezca sea la pag index que es lo que los clientes deberían de ver, y en caso de ser un desarrollador pues apuntarías al 8080 para gestionar ya que conoces de su existencia

De igual forma probaremos su uso con datos móviles y nos fijaremos como apunta al 80 por defecto



11. Instalación de certificados en apache, nginx y node

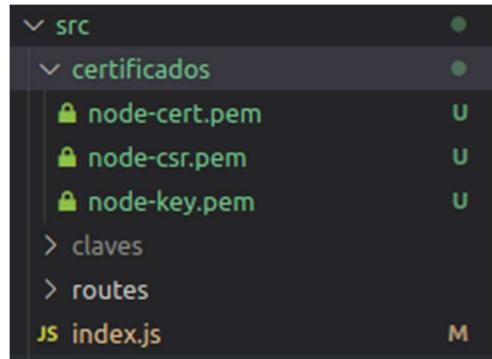
De forma profesional es evidente que deberíamos de adquirir un certificado validado y expedido por una entidad certificadora de confianza, pero como no es el caso, y supone un gasto el cual no quiero, vamos a crear un certificado auto firmado para nuestro servidor web apache y otro para las demás

11.1. Nginx(docker)

Crearemos sus certificados específicos

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes/src/certificados$ rm ryans-key.pem
mark@torre-ubuntu:~/Escritorio/pag-kubernetes/src/certificados$ openssl genrsa -out node-key.pem 1024
Generating RSA private key, 1024 bit long modulus (2 primes)
.....+++++
e is 65537 (0x010001)
mark@torre-ubuntu:~/Escritorio/pag-kubernetes/src/certificados$ openssl req -new -key node-key.pem -out node-csr.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.

mark@torre-ubuntu:~/Escritorio/pag-kubernetes/src/certificados$ openssl x509 -req -in node-csr.pem -signkey node-key.pem -out node-cert.pem
Signature ok
subject=C = es, ST = granada, L = santafe, O = mark, CN = mark
Getting Private key
```



Para securizar nginx de los contenedores docker lo que hare será crear a partir de la imagen de nginx de alpine una imagen que este securizada y para esto modiflico el dockerfile de ambos

```
FROM nginx:stable-alpine
COPY /asador/build/ /usr/share/nginx/html/
COPY default.conf /etc/nginx/conf.d/
COPY ./src/certificados/node-cert.pem /etc/ssl/
COPY ./src/certificados/node-key.pem /etc/ssl/

EXPOSE 80
EXPOSE 443

CMD ["nginx", "-g", "daemon off;"]
```

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile -t markblazquez/frontsec:v10 .
Sending build context to Docker daemon 697.4MB
Step 1/8 : FROM nginx:stable-alpine
--> 5c05ca045835
Step 2/8 : COPY /asador/build/ /usr/share/nginx/html
```

Dejamos así la conf

```
server {
    listen      80;
    listen  443 ssl;
    server_name  localhost;
    ssl_certificate      /etc/ssl/node-cert.pem;
    ssl_certificate_key /etc/ssl/node-key.pem;

    #access_log  /var/log/nginx/host.access.log  main;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }
}
```

Y como vemos ya funciona



Pero lo que voy a hacer es crear la redirección del puerto 80 al 443 para evitar problemas, así que repito los pasos de nuevo

```
server {
    listen 80 default_server;
    return 301 https://$host$request_uri;
}
server {
    listen 443 ssl;
    server_name  localhost;
    ssl_certificate      /etc/ssl/node-cert.pem;
    ssl_certificate_key /etc/ssl/node-key.pem;
}
```

Y en el contenedor funciona de forma satisfactoria con esto ya podemos pasar esta imagen a los deployment

Llegada la hora de pasarlo a kubernetes y si lo dejásemos así provocaría error por conflicto de puertos así que hay que habilitar el nuevo puerto en ambos servicios

```

    - name: dos
      protocol: TCP
      containerPort: 443

---
apiVersion: v1
kind: Service
metadata:
  annotations:
    name: nginx-demo-service
spec:
  ports:
    - name: uno
      port: 80
      targetPort: 80
      protocol: TCP
    - name: dos
      port: 443
      targetPort: 443
      protocol: TCP

```

Esto nos proporciona un nuevo puerto para app en 443, el cual vamos a ver en mi maravillosa app

Service						
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE	
kubernetes	ClusterIP	10.96.0.1		443/TCP	64d	
nginx-demo-service	NodePort	10.107.134.113		80:31952/TCP,443:30852/TCP	30d	

Ese puerto ahora lo debemos de añadir al portforward porque será donde redireccione de un lado a otro

<input checked="" type="checkbox"/>	pag asador index	TCP	*	192.168.1.132	80	31952	 
<input checked="" type="checkbox"/>	pag asador index ssl	TCP	*	192.168.1.132	443	30852	 

Y ahora sí que funciona la redirección al menos en el servicio de la página index, ya que el otro servicio para la página de gestión debemos de ajustar a los puertos 81 y 444 el problema es que nginx para el puerto por defecto de https sea diferente al 443 primero debe de pasar por este y se hace redirección al otro, quedaría tal que así redireccionando del puerto 81 al 443 y del 443 al 444

```

server {
    listen 81 default_server;
    return 301 https://$host$request_uri;
#return 301 $scheme://myexample.com:444$request_uri;

}server [
    listen 443 ssl ;
    server_name localhost;
    ssl_certificate      /etc/ssl/node-cert.pem;
    ssl_certificate_key /etc/ssl/node-key.pem;

    return 301 $scheme://$host:444$request_uri;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }

    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }

}

server {
    listen 444 ssl ;
    server_name localhost;
    ssl_certificate      /etc/ssl/node-cert.pem;
    ssl_certificate_key /etc/ssl/node-key.pem;
}

```

En mi caso esto es un problema porque como en el 443 ya tenemos una redirección a un puerto interno de la maquina da error así que lo más eficaz en este caso es pasar a apache que si permite cambiar de puerto sin redirecciones extrañas

11.2. Apache

11.2.1. Docker

Mirando la documentación del contenedor de apache nos da unas recomendaciones, como hacer este comando para sacar la conf, lo que me permitirá cambiar los puertos y hacerlo seguro

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker run --rm httpd:2.4 cat /usr/local/apache2/conf/httpd.conf > my-httpd.conf
```

Habilitar ssl des comentando estas líneas

```
...
#LoadModule socache_shmcb_module modules/mod_socache_shmcb.so
...
#LoadModule ssl_module modules/mod_ssl.so
...
#Include conf/extra/httpd-ssl.conf
...
```

Y defino los puertos que voy a utilizar y creo la redirección a la https

```
Listen 81
<VirtualHost _default_:81>
#ServerName torre-ubuntu.ddns.net
#Redirect permanent / https://torre-ubuntu.ddns.net

Redirect permanent / https://torre-ubuntu.ddns.net:444/
</VirtualHost>
```

Ahora es momento de modificar los ajustes de cuándo va por ssl por lo que saco la conf

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker run --rm httpd:2.4 cat /usr/local/apache2/conf/extra/httpd-ssl.conf > httpd-ssl.conf
```

Modifico puertos y los certificados

```
Listen 444
```

```
<VirtualHost _default_:444>
```

```
SSLCertificateFile "/usr/local/apache2/conf/server.crt.pem"
#SSLCertificateFile "/usr/local/apache2/conf/server-dsa.crt"
#SSLCertificateFile "/usr/local/apache2/conf/server-ecc.crt"

# Server Private Key:
# If the key is not combined with the certificate, use this
# directive to point at the key file. Keep in mind that if
# you've both a RSA and a DSA private key you can configure
# both in parallel (to also allow the use of DSA ciphers, etc.)
# ECC keys, when in use, can also be configured in parallel
SSLCertificateKeyFile "/usr/local/apache2/conf/server.key.pem"
#SSLCertificateKeyFile "/usr/local/apache2/conf/server-dsa.key"
```

Hago el dockerfile

```
FROM httpd:2.4
COPY /gestion/build/ /usr/local/apache2/htdocs/
COPY my-htpd.conf /usr/local/apache2/conf/httpd.conf
COPY httpd-ssl.conf /usr/local/apache2/conf/extra/
COPY ./src/certificados/node-cert.pem /usr/local/apache2/conf/server.crt.pem
COPY ./src/certificados/node-key.pem /usr/local/apache2/conf/server.key.pem

EXPOSE 81
EXPOSE 444
```

Creo la imagen y modiflico deployment

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker push markblazquez/frontsecapa:v34
The push refers to repository [docker.io/markblazquez/frontsecapa]
```

```
image: markblazquez/frontsecapa:v34
imagePullPolicy: IfNotPresent
ports:
  - name: uno
    protocol: TCP
    containerPort: 81
  - name: dos
    protocol: TCP
    containerPort: 444

---
apiVersion: v1
kind: Service
metadata:
  annotations:
    name: nginx-demo-service2
spec:
  ports:
    - name: uno
      port: 81
      targetPort: 81
      protocol: TCP
    - name: dos
      port: 444
      targetPort: 444
      protocol: TCP
```

Y por último modiflico el port forward ajustando a los puertos de este servicio

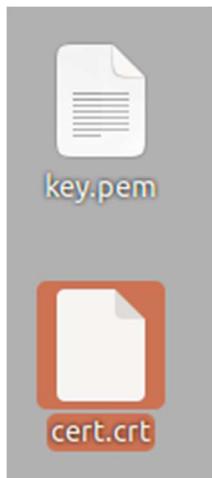
nginx-	NodePort	10.109.151.31	81:31317/TCP,444:32045/TCP	30d
demo-				
service2				

ON <input checked="" type="checkbox"/>	pagestionadasssl	TCP	*	192.168.1.132	444	32045	 
ON <input checked="" type="checkbox"/>	pagestionasador	TCP	*	192.168.1.132	81	31317	 

11.2.2. Servidor

Auto firmaremos nuestro servidor apache en la maquina, para ello introducimos este comando que lo que va a hacer es crear un certificado para apache con nuestra clave que también se crea aquí

```
mark@torre-ubuntu:~/Escritorio$ openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.crt -sha256 -days 365
Generating a RSA private key
.....+=====
writing new private key to 'key.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:es
State or Province Name (full name) [Some-State]:santafe
Locality Name (eg, city) []:santa
Organization Name (eg, company) [Internet Widgits Pty Ltd]:mark
Organizational Unit Name (eg, section) []:mark
Common Name (e.g. server FQDN or YOUR name) []:mark
Email Address []:
```

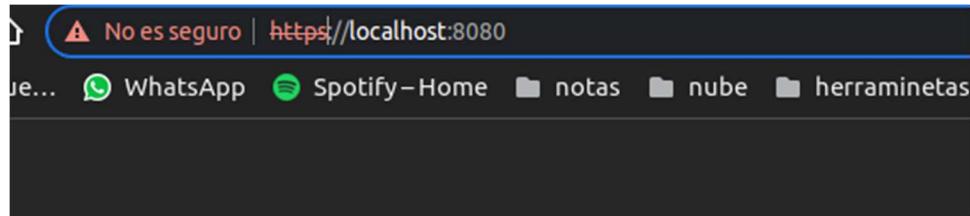


Una vez hecho esto ya podemos pasar a la config de apache

```
SSLEngine on
SSLCertificateFile      /home/mark/Escritorio/cert.crt
SSLCertificateKeyFile   /home/mark/Escritorio/key.pem
```



```
Mark@torre-ubuntu:/etc/apache2$ sudo a2enmod ssl
sudo a2enmod headers
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
Enabling module headers.
To activate the new configuration, you need to run:
  systemctl restart apache2
```



Y con esto como vemos la página ya tiene instalado un certificado con lo que nuestra página ya es segura frente a esnifados que es lo que nos interesa, para que el procesado de datos para acceso a la base de datos no sea en texto plano, sino que este cifrado y no se puedan obtener la información

Visor de certificados: mark

General Detalles

Enviado a

Nombre común (CN)	mark
Organización (O)	mark
Unidad organizativa (OU)	mark

Emitido por

Nombre común (CN)	mark
Organización (O)	mark
Unidad organizativa (OU)	mark

Período de validez

Emitido el	sábado, 21 de mayo de 2022, 19:53:47
Vencimiento el	domingo, 21 de mayo de 2023, 19:53:47

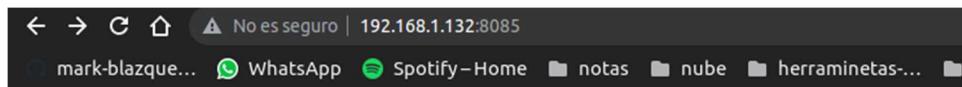
Huellas digitales

Huella digital SHA-256	D6 7F 93 F3 EC 0D 93 8B E4 98 F5 49 25 52 36 E5 0B 1E 41 C8 3E 34 8F 7A A7 B5 56 2D 6A FA A8 9E
Huella digital SHA-1	65 C0 49 48 F8 D5 E4 17 4F 2F 04 98 F1 2B 60 C4 74 EA 92 91

Igual para nuestra página de información de los servicios

```
mark@torre-ubuntu:/etc/apache2$ sudo nano sites-available/servicios.conf
mark@torre-ubuntu:/etc/apache2$ sudo systemctl restart apache2
sudo: systemctl: orden no encontrada
mark@torre-ubuntu:/etc/apache2$ sudo systemctl restart apache2
SSL Enter passphrase for SSL/TLS keys for kube.com:8085 (RSA): ****
```

Podemos ver como provoca fallo si vamos sin https, y esto es porque no he contemplado ofrecerla sin esto o hacer la redirección de la de sin certificados a la que sí , que tampoco lo voy a hacer

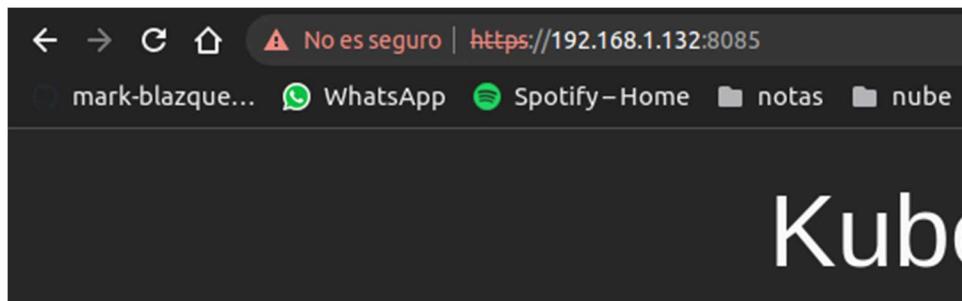


Bad Request

Your browser sent a request that this server could not understand.
Reason: You're speaking plain HTTP to an SSL-enabled server port.
Instead use the HTTPS scheme to access this URL, please.

Apache/2.4.48 (Ubuntu) Server at kube.com Port 8085

Y en cuanto vamos por https funciona



Pero por problemas de última hora con el router, cambio la app de login al puerto 8888 en apache e igual en el router

11.3. Node

Para el inicio de sesión de google dependeremos de node, al cual necesitamos añadir las rutas a la consola de google y hasta el momento se había trabajado de forma local por lo que funcionaba de forma correcta, pero al llevarlo a kubernetes recordemos que la entrada seria la ip del servicio por la cual accedemos que es un 10.x.x.x y ahora que ya vamos por nombre de dominio y accedemos por el port forward debemos de añadir nuestro dominio y si queremos añadir esto en google se nos presentara un error y es que necesita de certificados para acceder por lo que debemos hacer de nuestro servidor node un lugar más seguro con certificados

Empezamos instalando el módulo que se va a encargar de hacer esto

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ npm install https
added 1 package, and audited 238 packages in 2s
21 packages are looking for funding
  run `npm fund` for details
3 vulnerabilities (2 high, 1 critical)

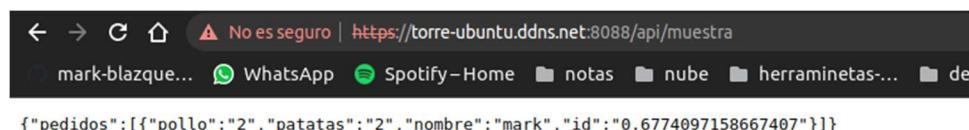
To address all issues, run:
  npm audit fix
Run `npm audit` for details.
```

Le pasamos al anterior método de creación del servidor el método de lectura para leer de los certificados

```
//modulo https
var https = require('https');

//creacion del servidor ahora con certificados
https.createServer({
  cert: fs.readFileSync('./certificados/node-cert.pem'),
  key: fs.readFileSync('./certificados/node-key.pem')
},app).listen(app.get('port'),()=>{
  console.log ("servidor corriendo en el puerto 8080")
})
```

Probamos que funcione de forma local



Y efectivamente funciona

Con esto ya solo falta añadir las nuevas direcciones que apuntaran al puerto de nuestro router que apunta al puerto de nuestro servicio kubernetes que ofrece el servidor node y otras dos rutas que son las pruebas en local

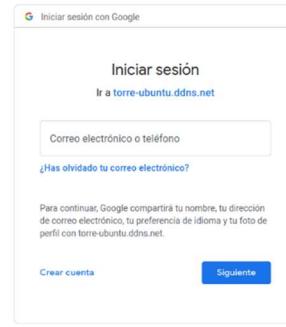
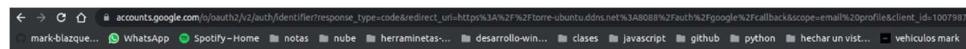
URI de redireccionamiento autorizados ?

Para usar con solicitudes de un servidor web

URI 1 *	<input type="text" value="https://torre-ubuntu.ddns.net:31059/auth/google/callback"/>
URI 2 *	<input type="text" value="https://torre-ubuntu.ddns.net:31059/api/auth/google"/>
URI 3 *	<input type="text" value="https://torre-ubuntu.ddns.net:8088/auth/google/callback"/>
URI 4 *	<input type="text" value="https://torre-ubuntu.ddns.net:8088/api/auth/google"/>

[+ AGREGAR URI](#)

Ahora revisaremos si el login se permite utilizando el dns y ssl



Con eso ya tendríamos nuestro servidor node securizado y a google permitiéndonos iniciar sesión al ser este un servidor seguro

Es momento de probar todo de forma definitiva así que con todos los cambios en código me dispongo a actualizar todas las imágenes de docker que utiliza kubernetes

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile2 -t markblazquez/contfront:v200 .
Sending build context to Docker daemon 696.9MB
Step 1/4 : FROM nginx:stable-alpine
--> 5c05ca045835
Step 2/4 : COPY ./gestion/build/ /usr/share/nginx/html
--> 3c7b42d1be80
Step 3/4 : EXPOSE 80
--> Running in 03f64e5af8d6
Removing intermediate container 03f64e5af8d6
--> d6a60717d4cc
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
--> Running in b9ae360dd099
Removing intermediate container b9ae360dd099
--> 4ba7360628ef
Successfully built 4ba7360628ef
Successfully tagged markblazquez/contfront:v200
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile -t markblazquez/contfront:v100 .
```

Y para la de node actualizamos el dockerfile y le pasamos a la imagen que ya sabemos que funciona lo nuevo actualizado

```
FROM markblazquez/backendk
COPY ./src /var/pag-kubernetes/
WORKDIR /var/pag-kubernetes/
EXPOSE 8080
CMD [ "node", "src/index.js" ]
```

Esto no funciona porque he tenido que adaptar un par de cosas

```
mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile -t markblazquez/backendk:100 .
Sending build context to Docker daemon 696.9MB
Step 1/4 : FROM nginx:stable-alpine
--> 5c05ca045835
Step 2/4 : COPY ./asador/build/ /usr/share/nginx/html
--> Using cache
--> 2b56ea63c220
Step 3/4 : EXPOSE 80
--> Using cache
--> a5b34c71b266
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
--> Using cache
--> 88118dad6abb
Successfully built 88118dad6abb
Successfully tagged markblazquez/backendk:100
```

Como dato curioso mencionar que el certificado da un error de longitud cuando el servidor node funciona sobre docker se soluciona haciéndolo de 2048

```
root@134a03b1461c:/var/pag-kubernetes# node src/index.js
_node_diagnostics.js:129
    c.context.setCert(cert);
    ^
Error: error:140AB18F:SSL routines:SSL_CTX_use_certificate:ee key too small
    at Object.createSecureContext (_node_diagnostics.js:129:17)
    at Server.setSecureContext (_tls_wrap.js:1328:27)
    at Server (_tls_wrap.js:1186:8)
    at new Server (https.js:70:14)
    at Object.createServer (https.js:94:10)
    at Object.<anonymous> (/var/pag-kubernetes/src/index.js:169:7)
    at Module._compile (internal/modules/cjs/loader.js:999:30)
    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1027:10)
    at Module.load (internal/modules/cjs/loader.js:863:32)
    at Function.Module._load (internal/modules/cjs/loader.js:708:14) {
  library: 'SSL routines',
  function: 'SSL_CTX_use_certificate',
  reason: 'ee key too small',
  code: 'ERR_SSL_EE_KEY_TOO_SMALL'
}
```

```
root@134a03b1461c:/var/pag-kubernetes/src/certificados# openssl genrsa -out node-key.pem 2048
```

```

{"pedidos":[{"pollo":"2","patatas":"2","nombre":"mark","id":"0.6774097158667407"}]}

```

```

mark@torre-ubuntu:~/Escritorio/pag-kubernetes$ docker build -f Dockerfile3 -t markblazquez/backendk:110 .
Sending build context to Docker daemon 697.3MB
Step 1/4 : FROM markblazquez/backendk
--> dc8868f26ae0
Step 2/4 : WORKDIR /var/pag-kubernetes/
--> Running in 427d91b766c3
Removing intermediate container 427d91b766c3
--> 1b9b2f68764a
Step 3/4 : EXPOSE 8080
--> Running in dbd86309091c
Removing intermediate container dbd86309091c
--> 119a3c1af98f
Step 4/4 : CMD [ "node", "src/index.js" ]
--> Running in 24bc92a88044
Removing intermediate container 24bc92a88044
--> 83c04fcfc1d2
Successfully built 83c04fcfc1d2
Successfully tagged markblazquez/backendk:110

```

Resulta curioso que cuando realizo la prueba me vuelca un fallo para obtener el acces token, y esto en local no pasa, me imagino que es una herramienta de google al servirse de forma publica

<https://stackoverflow.com/questions/21129989/internaloautherror-failed-to-obtain-access-token>

Y como en esta página nos indica es posible que sea debido por el certificado auto firmado, no obstante, añadimos la línea de código que se aconseja para evitar esto

```

InternalOAuthError: Failed to obtain access token
at Strategy._OAuth2Strategy._createAuthError (/var/pag-kubernetes/node_modules/passport-oauth2/lib/strategy.js:423:17)
at /var/pag-kubernetes/node_modules/passport-oauth2/lib/strategy.js:177:45
at /var/pag-kubernetes/node_modules/oauth/lib/oauth2.js:191:18
at ClientRequest.<anonymous> (/var/pag-kubernetes/node_modules/oauth/lib/oauth2.js:162:5)
at ClientRequest.emit (node:events:527:28)
at TLSSocket.socketErrorListener (node: http_client:454:9)
at TLSSocket.emit (node:events:527:28)
at emitErrorNT (node:internal/streams/destroy:157:8)
at emitErrorCloseNT (node:internal/streams/destroy:122:3)
at processTicksAndRejections (node:internal/process/task_queues:83:21)

```

Después de revisar no consigo arreglar el error del acces token por lo que me decanto por lo más básico ¿tiene acceso ese pod a internet?, y efectivamente después de hacer una conexión veo que no

```

mark@torre-ubuntu:~$ kubectl exec -it node-7d96d57c7d-tszh6 -- /bin/bash
root@node-7d96d57c7d-tszh6:/var/pag-kubernetes# apt update
Err:1 http://deb.debian.org/debian bullseye InRelease
  Temporary failure resolving 'deb.debian.org'
Err:2 http://security.debian.org/debian-security bullseye-security InRelease
  Temporary failure resolving 'security.debian.org'
0% [Working]^C

```

A falta de una semana para la entrega la solución pasa por eliminar la estructura del cluster y añadir esta vez la conexión de calico, la cual ya me dio error en un principio, por lo que asumiré este error y no le aplicare una solución

12. Instalación y configuración de firewall

De la forma en la que tenemos configurada ahora mismo nuestra red cualquier petición externa a nuestro dns apuntando a un puerto deseado será aceptada, a excepción de unos predefinidas por el router que serán bloqueadas o re direccionadas a otros puertos que yo he definido (el port forward), pero es evidente que esto puede ser una brecha de seguridad ya que no queremos que puedan acceder a pruebas de páginas web u otros tipos de servicios que se pueden estar ofreciendo de manera local por ejemplo nuestro servidor de base de datos.

Así que para ello lo me voy a valer de un firewall que cierre todos los puertos menos los que a mí me interesen, dejando solo así accesible los servicios por el port forward del router

Para ello me voy a valer de ufw que básicamente es iptables de forma asequible, así que lo instalo

```
mark@torre-ubuntu:~$ sudo apt install ufw  
[sudo] contraseña para mark:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias... Hecho  
Resolviendo dependencias... Hecho
```

Lo primero será activarlo y conocer que ufw de serie cierra todos los puertos, por lo que si hacemos una buscada de nuestra página web podemos ver que está bloqueada, así que es hora de ir añadiendo puertos, en concreto los internal port de la lista del port forward del rúter que son los que quiero que se tenga acceso por parte de los clientes

```
mark@torre-ubuntu:~$ sudo ufw enable  
El cortafuegos está activo y habilitado en el arranque del sistema
```

External Port	Internal Port
22	22
8888	8888
8085	8085
80	31952
443	30852
444	32045
81	31317
31059	31059

```
mark@torre-ubuntu:~$ sudo ufw allow 22  
Regla añadida  
Regla añadida (v6)  
mark@torre-ubuntu:~$ sudo ufw allow 8888  
Regla añadida  
Regla añadida (v6)  
mark@torre-ubuntu:~$ sudo ufw allow 8085  
Regla añadida  
Regla añadida (v6)  
mark@torre-ubuntu:~$ sudo ufw allow 31952  
Regla añadida  
Regla añadida (v6)  
mark@torre-ubuntu:~$ sudo ufw allow 30852  
Regla añadida  
Regla añadida (v6)  
mark@torre-ubuntu:~$ sudo ufw allow 31317  
Regla añadida  
Regla añadida (v6)  
mark@torre-ubuntu:~$ sudo ufw allow 31059  
Regla añadida  
Regla añadida (v6)
```

Así es como conseguimos por ejemplo tener nuestra página en desarrollo en el puerto 86 solo accesible para nosotros y no terceros cosa que antes no pasaba (la vuelvo a poner usable porque será un ejemplo de nginx stop para symphony)



Con esto nuestras páginas hospedadas en kubernetes no funcionarían ya que la conexión del otro equipo se perdería así que lo que hago es permitir todos los puertos a la ip del otro servidor

```
mark@torre-ubuntu:~$ sudo ufw allow from 192.168.1.130  
Regla actualizada
```

Con esto el error es subsanado y tenemos acceso a nuestra página web

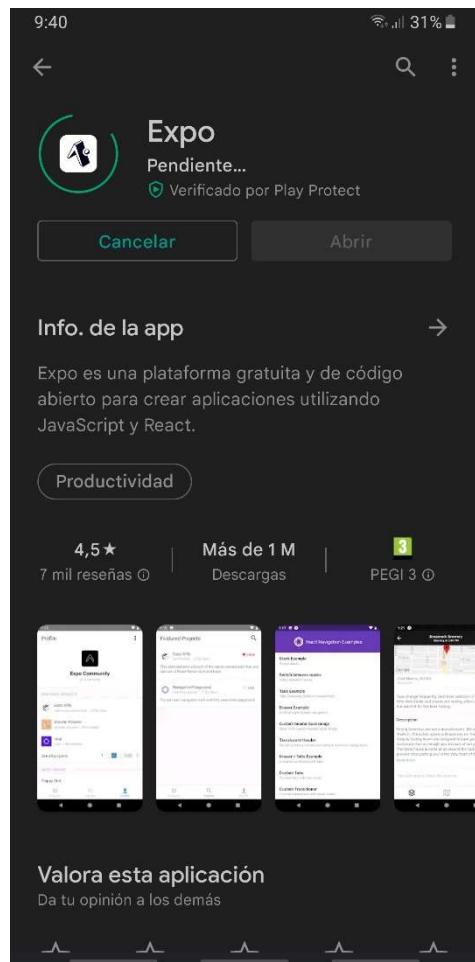
13. Port de la aplicación a dispositivos móviles

13.1. Explicación del entorno

Comenzamos con la instalación del entorno de programación para Android en el equipo

```
markdomen-17:~/Escritorio/expo$ npm install --global expo-cli
npm WARN deprecated source-map-url@0.4.1: See https://github.com/lydell/source-map-url#deprecated
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated source-map-resolve@0.5.3: See https://github.com/lydell/source-map-resolve#deprecated
npm WARN deprecated chokidar@2.1.8: Chokidar 2 does not receive security updates since 2019. Upgrade to chokidar 3 with a major version number change to break your dependency on its API and review the migration to prevent breaking changes. For more information, see https://github.com/lydell/chokidar#3-0-changes
npm WARN deprecated chokidar@2.1.8: Chokidar 2 does not receive security updates since 2019. Upgrade to chokidar 3 with a major version number change to break your dependency on its API and review the migration to prevent breaking changes. For more information, see https://github.com/lydell/chokidar#3-0-changes
npm WARN deprecated querystring@0.2.0: The querystring API is considered Legacy. new code should use the URLSearchParams API instead.
npm WARN deprecated uidv4@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() details.
npm WARN deprecated uidv4@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() details.
npm WARN deprecated uidv4@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() details.
```

En el móvil instalamos expo



Creamos el proyecto

```

mark@omen-17:~/Escritorio/expo2$ expo init app-movil-asador
✓ Choose a template: > tabs (TypeScript)    several example screens and tabs using react-navigation
✓ Downloaded template.
✖ Using Yarn to install packages. Pass --npm to use npm instead.
✓ Installed JavaScript dependencies.

✅ Your project is ready!

To run your project, navigate to the directory and run one of the following yarn commands.

- cd app-movil-asador
- yarn start # you can open iOS, Android, or web from here, or run them directly with the command
- yarn android
- yarn ios # requires an iOS device or macOS for access to an iOS simulator
- yarn web
mark@omen-17:~/Escritorio/expo2$ cd app-movil-asador/
mark@omen-17:~/Escritorio/expo2/app-movil-asador$ yarn start
yarn run v1.22.18
$ expo start
Starting project at /home/mark/Escritorio/expo2/app-movil-asador
Developer tools running on http://localhost:19002
Starting Metro Bundler

```

Lo iniciamos

```

mark@omen-17:~/Escritorio/expo2/app-movil-asador$ yarn start
yarn run v1.22.18
$ expo start

There is a new version of expo-cli available (5.4.6).
You are currently using expo-cli 5.4.4
Install expo-cli globally using the package manager of your choice;
for example: `npm install -g expo-cli` to get the latest version

Starting project at /home/mark/Escritorio/expo2/app-movil-asador
Developer tools running on http://localhost:19002
Starting Metro Bundler

> Metro waiting on exp://192.168.2.30:19000
> Scan the QR code above with Expo Go (Android) or the Camera app (iOS)

```

Y desde el teléfono que en mi caso me voy a centrar en desarrollar la app para Android, escaneamos el código y se abrirá la vista de la app del móvil

```

export default function Navigation({ colorScheme }: { colorScheme: ColorSchemeName }) {
  return (
    <NavigationContainer
      linking={LinkingConfiguration}
      theme={(colorScheme === 'dark' ? DarkTheme : DefaultTheme)}
      >
      <RootNavigator />
    </NavigationContainer>
  );
}

/**
 * A root stack navigator is often used for displaying modals on top of all other content.
 * https://reactnavigation.org/docs/modal
 */
const Stack = createNativeStackNavigator<RootStackParamList>();

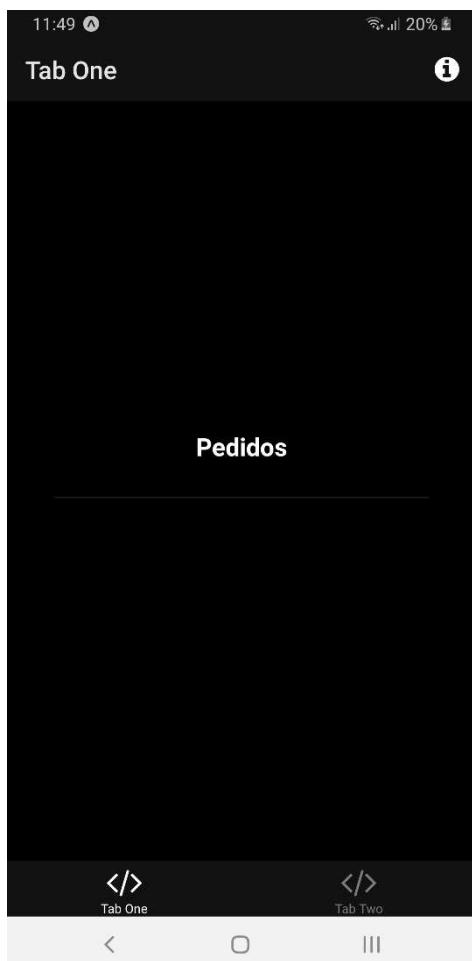
function RootNavigator() {
  return (
    <Stack.Navigator>
      <Stack.Screen name="Root" component={BottomTabNavigator} options={{ headerShown: false }} />
      <Stack.Screen name="NotFound" component={NotFoundScreen} options={{ title: 'Oops!' }} />
      <Stack.Screen name="Modal" component={ModalScreen} />
    </Stack.Navigator>
  );
}

/**
 * A bottom tab navigator displays tab buttons on the bottom of the display to switch screens.
 * https://reactnavigation.org/docs/bottom-tab-navigator
 */
const BottomTab = createBottomTabNavigator<RootTabParamList>();

function BottomTabNavigator() {
  const colorScheme = useColorScheme();

  return (
    <BottomTab.Navigator
      initialRouteName="TabOne"
      screenOptions={{
        tabBarActiveTintColor: colors[colorScheme].tint,
      }}
    >
      <BottomTab.Screen

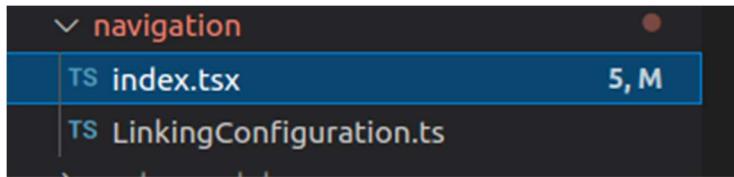
```



Así que procedo a desarrollar la app de gestión como app nativa para móvil y en este caso nos incluye un modelo con una vista de dos ventanas, en este caso se trabajara con typescript que no deja de ser javascript pero tipado por lo que a veces ayuda a identificar algunos fallos y en otras ocasiones no es nada más que un dolor de cabeza

Para empezar, visualizo y explico la estructura del proyecto

Navigation será el index donde se renderiza las screen y el menú de navegación



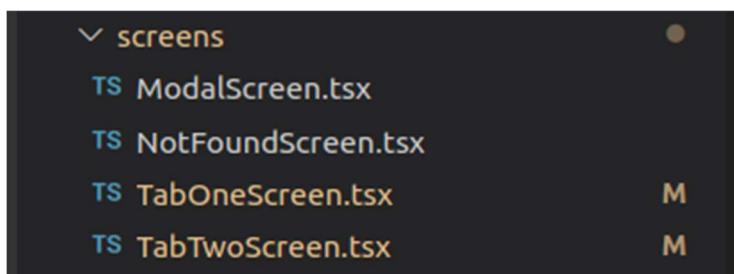
Esta función nos devolverá el menú de abajo donde podemos cambiar de pantalla

```
function BottomTabNavigator() {
  const colorScheme = useColorScheme();

  return (
    <BottomTab.Navigator
      initialRouteName="TabOne"
      screenOptions={{
        tabBarActiveTintColor: Colors[colorScheme].tint,
      }}>
      <BottomTab.Screen
        name="TabOne"
        component={TabOneScreen}
        options={({ navigation }: RootTabScreenProps<'TabOne'>) => ({
          title: 'Tab One',
          tabBarIcon: ({ color }) => <TabBarIcon name="code" color={color} />,
          headerRight: () => (
            <Pressable
              onPress={() => navigation.navigate('Modal')}
              style={({ pressed }) => ({
                opacity: pressed ? 0.5 : 1,
              })}
            >
              <FontAwesomeIcon
                name="info-circle"
                size={25}
                color={Colors[colorScheme].text}
                style={{ marginRight: 15 }}
              >
            
```



En screens irán las diferentes pantallas

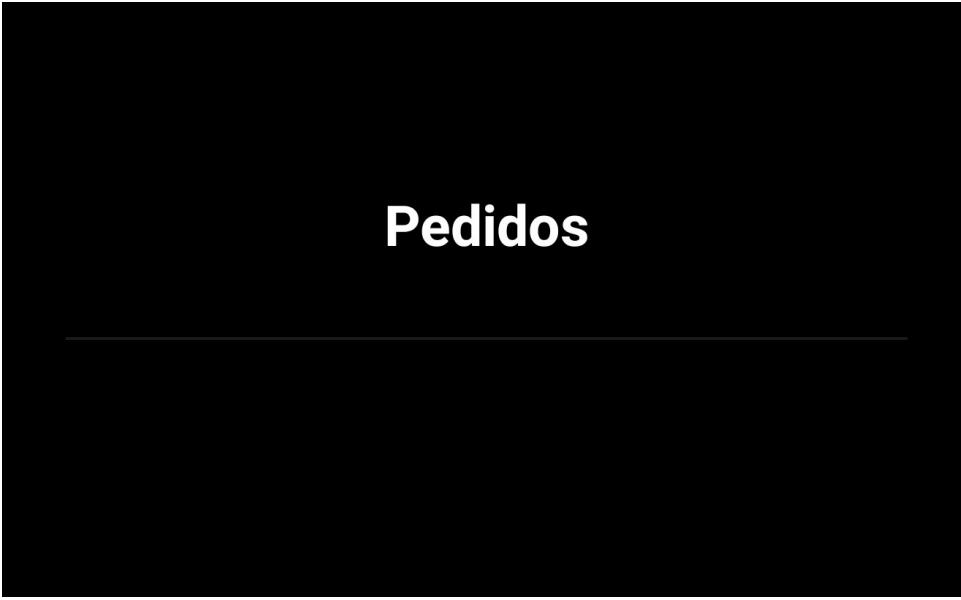


Esta será la pestaña donde se renderizan los pedidos del api

```
import EditScreenInfo from '../components/EditScreenInfo';
import { Text, View } from '../components/Themed';
import { RootTabScreenProps } from '../types';
import {load} from '../api'

export default function TabOneScreen({ navigation }: RootTabScreenProps<'TabOne'>) {
  return (
    <View style={styles.container}>
      <Text style={styles.title}>Pedidos</Text>
      <View style={styles.separator} lightColor="#eee" darkColor="rgba(255,255,255,0.7)">
        </View>
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    alignItems: 'center',
    justifyContent: 'center',
  },
  title: {
    fontSize: 20,
    fontWeight: 'bold',
  },
  separator: {
    width: '100%',
    height: 1,
    margin: 10,
  },
});
```



Pedidos

El siguiente paso es modificar lo necesario para obtener los pedidos desde el json

13.2. Mostrar datos de api

Para ello crearemos la función que hace una petición al servidor node y devolveremos los datos

```
const [isLoading, setLoading] = useState(true);
const [data, setData] = useState([]);

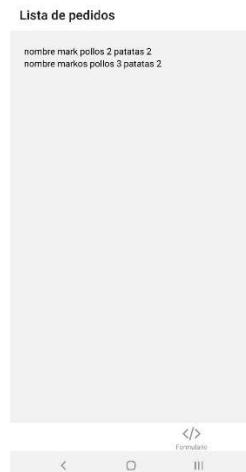
const getdata = async () => {
  try {
    const response = await fetch('http://192.168.2.30:8080/api/muestra');
    const json = await response.json();
    console.log(json)
    setData(json.pedidos);
  } catch (error) {
    console.error(error);
  } finally {
    setLoading(false);
  }
}

useEffect(() => {
  getdata();
}, []);
```

Vemos el useEffect que es un hook que ejecuta esta función cada vez que se recarga la página , y la parte de useState que lo que hace es definir un objeto vacío y lo que va a hacer es rellenarlo cada vez que es lo que nos permite guardar los datos y utilizarlos

```
return (
  <View style={{ flex: 1, padding: 24 }}>
    {isLoading ? <ActivityIndicator/> : (
      <FlatList
        data={data}
        renderItem={({ item }) => (
          <Text>nombre {item.nombre} pollos {item.pollo} patatas {item.patatas} </Text>
        )}
      />
    )}
  </View>
);
```

De esta forma es como lo mostramos por pantalla y el resultado sería el siguiente



Pasamos a la página 2 donde hay un formulario que servirá para la inserción de datos al servidor

13.3. Añadir un screen y su posición en el menú

Necesitare un screen más, así que partiendo del menú que nos da esta posibilidad debemos de añadir un par de cosillas como la definición de este objeto nuevo y su creación en el nivel inferior

```
TS types.tsx M X
```

```
export type RootTabParamList = {
  TabOne: undefined;
  TabTwo: undefined;
  Tabtres: undefined;
};
```

```
s LinkingConfiguration.ts M X
```

```
,,
  Tabtres: {
    screens: {
      TabtresScreen: 'tres',
    },
  },
},
```

```
TS index.tsx 5, M X
```

```
<BottomTab.Screen
  name="Tabtres"
  component={TabtresScreen}
  options={{
    title: 'Borrar',
    tabBarIcon: ({ color }) => <TabBarIcon name="code" color={color} />,
  }}
/>
```

13.4. Añadir

Definimos el formulario donde en esta ocasión el id lo introduciremos nosotros y será alfanumérico

```
return (
  <SafeAreaView>
    <TextInput
      style={styles.input}
      placeholder="nombre"
      //le pasa texto y sustitulle el objeto vacio por ese texto
      onChangeText={(text)=>handleChange('nombre',text)}

    />
    <TextInput
      style={styles.input}
      placeholder="pollo"
      keyboardType="numeric"
      onChangeText={(text)=>handleChange('pollo',text)}

    />
    <TextInput
      style={styles.input}
      placeholder="patatas"
      keyboardType="numeric"
      onChangeText={(text)=>handleChange('patatas',text)}

    />
    <TextInput
      style={styles.input}
      placeholder="id"
      onChangeText={(text)=>handleChange('id',text)}

    />
    <Button
      onPress={handleSubmit}
      title="enviar"
    />

  </SafeAreaView>
```

Añadir

The screenshot shows a mobile application interface. At the top, there is a header labeled "Añadir". Below it is a form consisting of four input fields: "nombre", "pollo", "patatas", and "id". Each field has a placeholder text inside. Below the inputs is a blue button with the text "ENVIAR" in white. At the very bottom of the screen, there is a navigation bar with several icons and text labels: "</>" and "Lista de pedidos" on the left, and "</>" and "Borrar" on the right.

Es hora de hacer que lo del formulario pase a la función como objeto

```
//crea el objeto vacio
const [data ,setdata] = useState([
  nombre:"",
  pollo:"",
  patatas:"",
  id: ""
])

//actualiza los datos vacios pasando los datos del form
const handleChange = (name:any,value:any) => setdata({...data, [name]:value})

//lo guarda en el backend

const handleSubmit= () =>{
  //console.log(data)
  saveData(data)
```

Donde savedata que se encarga de guardar el archivo está en otro lado por limpieza y pasa la data definida antes como string a traves de cabeceras

```
export const saveData = async (newData) => {
  //const res = await fetch('https://torre-ubuntu.ddns.net:31059/api/nuevo', {
  const res = await fetch('http://192.168.68.113:8080/api/nuevo', {
    //const res = await fetch('http://192.168.2.30:8080/api/nuevo', {
      method: 'POST',
      headers: {
        'accept': 'application/json',
        'content-type': 'application/json',
        Authorization: JSON.stringify(newData)
      },
      //body:JSON.stringify(newData)
    });

    return await res.json()
}
```

Cambiamos el api para que, si existe la cabecera, en caso de trabajar desde móvil recoja los datos de ahí, y los convierta a objeto para trabajar con ellos que para eso sirve el parse

```
//metodo post - pasando un json vacio - unude el pedido enviado del front end a los per
app.post('/api/nuevo',(req,res) => {
  if(req.headers.authorization){
    //añadir el de móvil
    const newpedidosocio = req.headers.authorization
    const newpedido=JSON.parse(newpedidosocio)
    pedidos.push(newpedido)
    //cojemos el json de epdidos y lo guardadmos como texto
    const json_pedidos = JSON.stringify(pedidos)
    fs.writeFileSync(path.join(__dirname,'pedidos.json'), json_pedidos,'utf-8')
  }else{
    //obtenemos el pedido del form
    //console.log(req.body)
    const newpedido = req.body
    //console.log(newpedido)
```

13.5. Eliminar

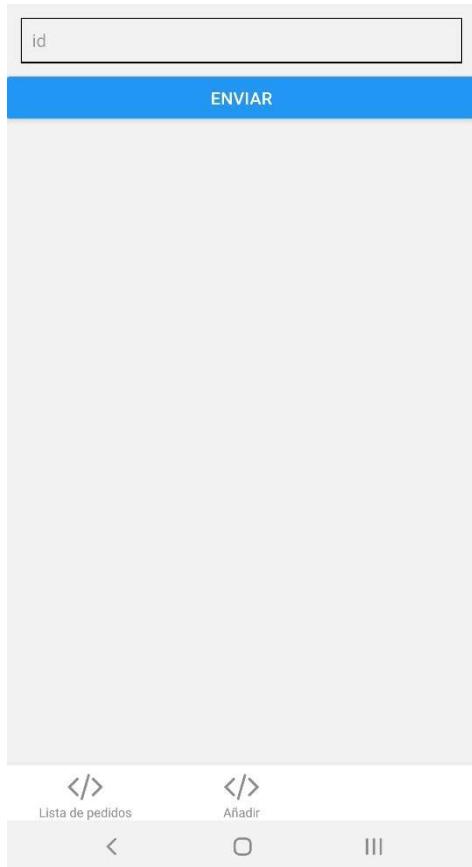
De igual forma que la inserción funciona la eliminación a través de formulario, pero en este caso se debe de proporcionar el id

```
return (
  <SafeAreaView>
    <TextInput
      style={styles.input}
      placeholder="id"
      keyboardType="numeric"
      //value= {Math.random()}
      onChangeText={(text)=>handleChange('id',text)}

    />
    <Button
      onPress={handleSubmit}
      title="enviar"
    />

  </SafeAreaView>
);
};
```

Borrar



Procesamiento de datos

```

const Form = () => {
    //crea el objeto vacio
    const [data ,setdata] = useState({
        id : ""
    })

    //actualiza los datos vacios pasando los datos del form
    const handleChange = (name:any,value:any) => setdata({...data, [name]:value})

    //lo borra en el backend
    const handleSubmit= () =>[
        //console.log(data)
        deleteData(data)
    ]
}

```

Envío de datos

```

//funcion borrado
export const deleteData = async (newData) => {
    //const res = await fetch('https://torre-ubuntu.ddns.net:31059/api/delete', {
    const res = await fetch('https://192.168.68.113:8080/api/delete', {
    //const res = await fetch('http://192.168.2.30:8080/api/delete', {
        method: 'POST',
        headers: {
            'accept': 'application/json',
            'content-type': 'application/json',
            Authorization: JSON.stringify(newData)
        },
        //body:JSON.stringify(newData)
    });

    return await res.json()
}

```

Cambiamos el api

```

app.post('/api/delete',(req,res) => {
    if(req.headers.authorization){
        //eliminar el de movil
        const cabecera = req.headers.authorization
        const eliminar=JSON.parse(cabecera)
        //console.log(eliminar)
        //const id=JSON.parse(newpedidosucio)
        pedidos = pedidos.filter(pedidos=> pedidos.id != eliminar.id )
        //console.log(pedidos);
        //se codifica para poder escribirse en el fichero
        const json_pedidos = JSON.stringify(pedidos)
        //se pasa al fichero para guardar el nuevo resultado
        fs.writeFileSync(path.join(__dirname , 'pedidos.json'), json_pedidos,'utf-8')
    }else{

```

13.6. Añadiendo recarga de pagina

Para recargar la página de forma manual en react native debemos de añadir código

```
const wait = (timeout) => {
  return new Promise(resolve => setTimeout(resolve, timeout));
}

const App = () => {
  const [refreshing, setRefreshing] = React.useState(false);

  const onRefresh = React.useCallback(() => {
    setRefreshing(true);
    wait(2000).then(() => setRefreshing(false));
  }, []);

  return (
    <SafeAreaView style={styles.container}>
      <ScrollView
        contentContainerStyle={styles.scrollView}
        refreshControl={
          <RefreshControl
            refreshing={refreshing}
            onRefresh={getData}
          />
        }
      >
        <FlatList
          data={data}
          renderItem={({ item }) => (
            //extraccion de la info
            <Text>nombre {item.nombre} pollos {item.pollo} patatas {item.patatas} id {item.id}</Text>
          )}
        />
      </ScrollView>
    </SafeAreaView>
  );
}
```

Con esto ya podemos recargar los pedidos si tenemos la versión web y móvil funcionando a la vez que todo esté bien actualizado

13.7. Redirección de página cuando se hace el formulario

Para que cuando se presione el botón redirija a la página de pedidos

```
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/native-stack';

const Form = function HomeScreen({ navigation }) {

  const handleSubmit= () =>{
    //console.log(data)
    saveData(data)
    navigation.navigate('TabOne')
  }
}
```

13.8. Problemas con https (certificados node)

Debido a que mi servidor node es seguro, pero con certificados con una firma no de confianza cuando va por https produce un error por eso se puede ver en las imágenes que va por la opción no segura, lo más fácil ha sido expedir un certificado de confianza para node ya que react native es una tecnología muy nueva y errática falta de soluciones eficaces como el ignorar estos certificados que si se puede hacer, pero cuando se ha compilado como una aplicación para Android

Utilizare lets encrypt que proporciona unos certificados de forma instantánea y en principio los reconoce como confiables, aunque salte la alerta de que no lo son

```
mark@torre-ubuntu:~$ sudo certbot certonly --manual
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator manual, Installer None
Please enter in your domain name(s) (comma and/or space separated) (Enter 'c'
to cancel): torre-ubuntu.ddns.net
Requesting a certificate for torre-ubuntu.ddns.net
Performing the following challenges:
http-01 challenge for torre-ubuntu.ddns.net

-----
Create a file containing just this data:

H76Gr-b4K-D6BKD_U08Rso_UDSlRi_DLjRjGXqqzRw8.cFCV_D6_7p5bgtKH-eePVLPjbkDDYJ0EsFYdPEgj0E4

And make it available on your web server at this URL:

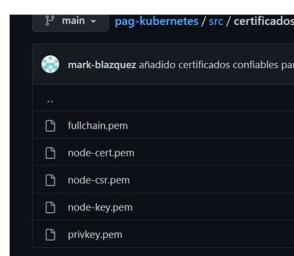
http://torre-ubuntu.ddns.net/.well-known/acme-challenge/H76Gr-b4K-D6BKD_U08Rso_UDSlRi_DLjRjGXqqzRw8

-----
Press Enter to Continue
Waiting for verification...
Cleaning up challenges

IMPORTANT NOTES:
- Congratulations! Your certificate and chain have been saved at:
  /etc/letsencrypt/live/torre-ubuntu.ddns.net/fullchain.pem
  Your key file has been saved at:
  /etc/letsencrypt/live/torre-ubuntu.ddns.net/privkey.pem
  Your certificate will expire on 2022-09-06. To obtain a new or
  tweaked version of this certificate in the future, simply run
  certbot again. To non-interactively renew *all* of your
  certificates, run "certbot renew"
- If you like Certbot, please consider supporting our work by:

  Donating to ISRG / Let's Encrypt:  https://letsencrypt.org/donate
  Donating to EFF:                 https://eff.org/donate-le
```

Se puede observar que aquí ya los crea y los guarda y ahora lo único que debemos de hacer es añadirlos al directorio de node y especificar estos nuevos certificados



```
https.createServer({
    cert: fs.readFileSync('src/certificados/fullchain.pem'),
    key: fs.readFileSync('src/certificados/privkey.pem'),
    rejectUnauthorized: false
},app).listen(app.get('port'),()=>{
    console.log ("servidor corriendo en el puerto 8080")
})
```

Con esto ya detecta que la pagina es segura y permite obtener los datos del servidor node hospedado en kubernetes de forma correcta