

TFG ASIR II

Proyecto Lorient

Junio 19,2024

Administración de Sistemas
Informáticos en Red



PROYECTO LORIENT

"AUTOMATIZACIÓN Y ESCALABILIDAD
PARA UN FUTURO SIN LÍMITES"

ÍNDICE

TFG ASIR II.....	1
Proyecto Lorient.....	1
—.....	1
—.....	1
ÍNDICE.....	3
INTRODUCCIÓN.....	5
Componentes del Proyecto.....	7
Azure Cloud:.....	7
Jenkins VM:	7
Terraform:	8
Ansible:.....	8
Prometheus y Grafana:	9
Balanceador de Carga:	10
Mongodb ReplicaSet:	10
Mongodb Replica Set Azure Locations:	10
Diagrama de flujo del proyecto:	11
JUSTIFICACIÓN.....	12
OBJETIVOS DEL PROYECTO.....	14
1. Automatización de Infraestructura:	14
2. Alta Disponibilidad:	14
3. Despliegue Redundante:	14
4. Eficiencia Operativa:	14
5. Monitorización Avanzada:	14
6. Escalabilidad:	14
7. Seguridad:	14
PLANIFICACIÓN/EXPLICACIÓN	15
2. Creación del directorio de Terraform:	16
Resource Groups:.....	18
1-LLA-TFG	18
2-LLA-TFG	19
3-LLA-TFG	19
3. Desarrollo de los playbooks de Ansible:	20
4. Configuración del Load Balancer:.....	20
5. Monitoreo con Prometheus y Grafana:.....	20
CONCLUSIONES	22

INTRODUCCIÓN

Hoy en día, en un entorno tecnológico donde la agilidad, eficiencia y escalabilidad son esenciales para el éxito de cualquier empresa u organización, el despliegue de infraestructura en la nube se ha convertido en una necesidad más que una opción.

Este proyecto tiene como objetivo diseñar e implementar una infraestructura en la nube utilizando herramientas de automatización y monitoreo para llevar la eficiencia, escalabilidad y disponibilidad de los servicios al siguiente nivel. Utilizaremos una combinación de varias tecnologías de automatización de alto nivel, como Jenkins, Terraform, Ansible, Prometheus y Grafana, todas alojadas en la plataforma de Microsoft Azure teniendo como finalidad desplegar un servidor de bases de datos MongoDB Replica SET.

Estas herramientas facilitarán nuestro trabajo diario al crear un entorno de automatización donde el despliegue de la infraestructura y su monitoreo se realizan con solo unos pocos clics. El objetivo principal es diseñar e implementar una infraestructura automatizada y escalable en la nube utilizando Terraform y Ansible. Esto permitirá crear, configurar y gestionar un clúster de MongoDB de manera eficiente y reproducible. La automatización reducirá demasiado el tiempo necesario para realizar el despliegue y la configuración de servicios, reduciendo los posibles errores humanos y garantizando la consistencia en entornos Linux más específicamente esta diseñado para Sistemas Operativos Ubuntu.

Para garantizar la alta disponibilidad y la tolerancia a fallos, el clúster de MongoDB se configurará como un replica set. Un replica set en MongoDB es un grupo de instancias de mongod que mantienen los mismos datos, proporcionando redundancia y aumentando la disponibilidad de los datos. En caso de que el servidor primario falle, uno de los servidores secundarios se promoverá automáticamente a primario, asegurando así la continuidad del servicio sin intervención manual.

Además, para asegurar la disponibilidad del servicio web, se desplegarán dos máquinas virtuales adicionales en diferentes ubicaciones de Azure. Esto permitirá que, en caso de que una de las ubicaciones falle, el servicio continúe operativo desde la otra ubicación. Un balanceador de carga distribuirá las peticiones entrantes entre los dos servidores, garantizando una distribución eficiente del tráfico y aumentando la resiliencia del sistema.

En resumen, este proyecto tiene como fin crear una infraestructura en la nube completamente automatizada y escalable. Utilizando Terraform para la provisión de la infraestructura y Ansible para la configuración de los servicios, se implementará un clúster de MongoDB con replica set y servidores web distribuidos en diferentes

ubicaciones. Esto garantizará la alta disponibilidad y la resiliencia del sistema, proporcionando un servicio confiable y eficiente.

Componentes del Proyecto

Azure Cloud:

- Utilizaremos la plataforma de Microsoft Azure para alojar nuestra infraestructura en la nube. Azure proporcionará los recursos necesarios, como máquinas virtuales, redes, y almacenamiento, para desplegar y operar nuestro sistema.

Jenkins VM:

- ¿Qué es Jenkins? Es un servidor open source para la integración continua. Se usa para probar y compilar proyectos de software de forma continua. Con Jenkins se facilita la incorporación de cambios en un desarrollo y la entrega de nuevas versiones.
- Servirá como la herramienta de automatización de pipelines de CI/CD. Jenkins gestionará los procesos de integración y despliegue continuo, asegurando que los cambios en el código y la infraestructura se implementen de manera eficiente y sin errores.

Usaremos un pipeline de tipo “Pipeline Script from SCM”, lo que hará será bajar un repositorio desde Gitlab donde tendrá su Pipeline y todo lo necesario para la ejecución de Terraform y Ansible.

T

TFG

main

TFG /


+

History

Find file

Edit

Code


Update Jenkinsfile
Lauti Lorca authored 1 week ago

4fc4fa7f

Name	Last commit	Last update
Terraform	Update instance.tf	2 weeks ago
scripts	permis	2 weeks ago
Jenkinsfile	Update Jenkinsfile	1 week ago
README.md	ultim	2 weeks ago

README.md

Proyecto de Fin de Grado ASIR - Lautaro

Bienvenido al repositorio de mi Trabajo de Fin de Grado (TFG) para el ciclo formativo de Administración de Sistemas Informáticos en Red (ASIR). Este proyecto ha sido desarrollado por Lautaro.

42 Commits

1 Branch

0 Tags

52.1 MiB Project Storage

README

+ Add LICENSE

+ Add CHANGELOG

+ Add CONTRIBUTING

+ Add Kubernetes cluster

+ Set up CI/CD

+ Add Wiki

+ Configure Integrations

Created on

May 21, 2024

Terraform:

- Terraform es una poderosa herramienta de infraestructura como código que permite a los equipos de TI y desarrolladores automatizar la creación y gestión de infraestructuras completas. Con Terraform, se puede definir, provisionar y actualizar recursos en la nube y en servidores locales usando código, lo que facilita la gestión de infraestructuras en múltiples nubes públicas y privadas. Esto asegura configuraciones consistentes, reproducibles y eficientes, ahorrando tiempo y reduciendo errores humanos.
- Servirá como la herramienta de automatización de pipelines de CI/CD. Jenkins gestionará los procesos de integración y despliegue continuo, asegurando que los cambios en el código y la infraestructura se implementen de manera eficiente y sin errores.

Ansible:

- Ansible es una herramienta que automatiza la configuración y administración de sistemas informáticos. Permite definir y aplicar configuraciones en servidores y dispositivos de red de manera eficiente, simplificando tareas repetitivas y mejorando la consistencia y seguridad de los entornos de TI.
- Empleado para la configuración automática de sistemas. A través de playbooks de Ansible, se instalarán y configurarán los servicios de bases de datos en las máquinas virtuales, garantizando la consistencia y rapidez en la configuración de entornos.

Prometheus y Grafana:

- Prometheus y Grafana son dos herramientas que se complementan para monitorizar y visualizar sistemas y aplicaciones:
 - **Prometheus** es un sistema de monitorización y alerta de código abierto diseñado para recopilar métricas de sistemas y servicios. Utiliza un modelo de datos basado en series temporales y permite consultar y visualizar estas métricas.
 - **Grafana** es una plataforma de visualización de métricas y análisis de datos de código abierto. Se integra con diversas fuentes de datos, incluido Prometheus, y permite crear cuadros de mando y gráficos interactivos para visualizar y analizar datos de métricas.

Cómo funcionan juntos:

- **Prometheus** recopila métricas de diferentes sistemas y servicios utilizando su propio servidor web y un modelo de almacenamiento de series temporales.
- **Grafana** se conecta a Prometheus como fuente de datos para consultar y visualizar estas métricas en cuadros de mando personalizados.
- Los usuarios pueden configurar alertas en Prometheus y visualizar estas alertas en Grafana, lo que proporciona una integración completa para monitorear y analizar el rendimiento de sistemas y aplicaciones.

Esta combinación permite a los equipos de operaciones y desarrollo supervisar el rendimiento de sus sistemas en tiempo real y analizar tendencias históricas para mejorar la eficiencia y la confiabilidad de sus aplicaciones.

- Estas herramientas se utilizarán para el monitoreo y la visualización de métricas. Prometheus recolectará datos de rendimiento y estado de los servicios, mientras que Grafana proporcionará paneles interactivos para visualizar estas métricas y facilitar el monitoreo en tiempo real.

Balanceador de Carga:

- El balanceador de carga se encargará de distribuir las peticiones de los clientes entre los servidores del MongoDB Replica Set. Al configurar un solo dominio o dirección IP para el balanceador de carga, éste redirigirá automáticamente las peticiones entrantes hacia los servidores disponibles en el Replica Set. Esto asegura que los clientes puedan acceder al servicio de base de datos de manera continua y sin interrupciones, incluso si uno de los servidores del Replica Set falla o está en mantenimiento.

Mongodb ReplicaSet:

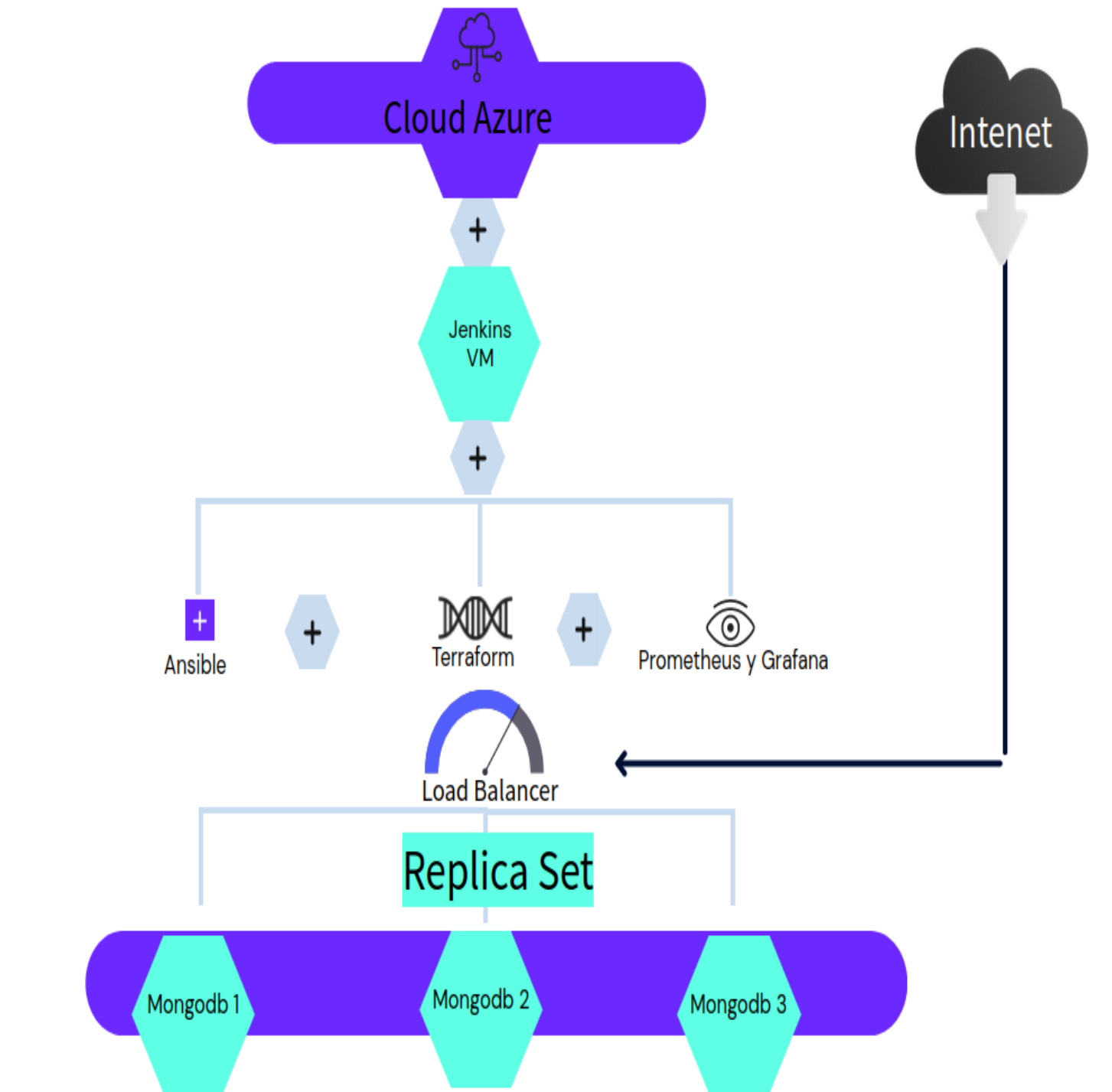
- Un MongoDB Replica Set es un grupo de servidores MongoDB que mantienen copias idénticas de los mismos datos para proporcionar alta disponibilidad y tolerancia a fallos. Consiste en un nodo primario que acepta operaciones de escritura y replica datos a nodos secundarios, que pueden ser configurados para permitir lecturas. En caso de fallo del nodo primario, un secundario se promueve automáticamente a primario, garantizando la continuidad del servicio.
- Se desplegará un clúster de MongoDB configurado como replica set para garantizar la alta disponibilidad y tolerancia a fallos. Un replica set en MongoDB es un conjunto de instancias de bases de datos que mantienen los mismos datos, asegurando redundancia y continuidad del servicio en caso de fallos.

Mongodb Replica Set Azure Locations:



REPLICASET
Locations

Diagrama de flujo del proyecto:



JUSTIFICACIÓN

Este proyecto surgió gracias a mis prácticas realizadas en Lorient. Cuando ingresé a la empresa y comencé a trabajar, aprendí y me interioricé día a día en su funcionamiento. Me di cuenta de que el nivel de automatización de la empresa era fenomenal. Siempre

que hablaba con algún superior, veía que su enfoque estaba centrado en la escalabilidad, ya que la empresa está creciendo exponencialmente y saben que no pueden realizar todo el trabajo ellos manualmente.

Por ejemplo, al ingresar un nuevo cliente, se crearía su entorno en alguna nube, creando las máquinas virtuales, desplegando su servicio y configurándolo a gusto del cliente. Este proceso llevaría demasiado tiempo si se hiciera manualmente. Por eso, utilizan tecnologías para desarrollar la automatización del despliegue, actualizaciones, etc. Con herramientas como Terraform, solo se necesita ejecutar un comando, como "terraform apply", para tener toda la infraestructura física creada en la nube. Luego, con Ansible, ejecutando un "ansible-playbook", pueden desplegar todo su servicio.

Un trabajo que antes tomaba muchas horas ahora se puede realizar con unos pocos comandos. Esta experiencia cambió mi mentalidad y me hizo pensar en función de la automatización y la escalabilidad. Así surgió la idea de este proyecto, donde utilizaremos estas tecnologías para desplegar un clúster de base de datos basado en un MongoDB replica set.

OBJETIVOS DEL PROYECTO

El proyecto tiene como objetivos principales:

1. **Automatización de Infraestructura:** Implementar un entorno de infraestructura automatizado en la nube utilizando Terraform y Ansible. Esto permitirá provisionar y configurar rápidamente los recursos necesarios para el despliegue de servicios.
2. **Alta Disponibilidad:** Configurar un clúster de MongoDB como replica set para garantizar la alta disponibilidad de los datos. Un replica set proporciona redundancia, permitiendo la promoción automática de servidores secundarios a primarios en caso de fallo.
3. **Despliegue Redundante:** Desplegar dos máquinas virtuales con servidores web en diferentes ubicaciones de Azure para asegurar la continuidad del servicio en caso de fallo de una ubicación. Un balanceador de carga distribuirá el tráfico entre ambos servidores.
4. **Eficiencia Operativa:** Reducir el tiempo y los errores asociados con el despliegue y la gestión de infraestructura y servicios mediante la automatización completa con herramientas como Terraform y Ansible.
5. **Monitorización Avanzada:** Implementar Prometheus y Grafana para la monitorización avanzada de la infraestructura y los servicios desplegados. Esto permitirá una visualización detallada del rendimiento y la salud del sistema en tiempo real.
6. **Escalabilidad:** Diseñar la infraestructura de manera que sea escalable, permitiendo el crecimiento y la expansión según las necesidades futuras de la empresa.
7. **Seguridad:** Implementar prácticas de seguridad recomendadas para proteger los datos y la infraestructura frente a amenazas y vulnerabilidades.

Estos objetivos se orientan hacia la mejora de la eficiencia operativa, la disponibilidad y la escalabilidad de los servicios, asegurando un entorno de nube robusto y confiable para la organización.

PLANIFICACIÓN/EXPLICACIÓN

La planificación y avance de este proyecto se basaron en un aprendizaje y práctica constantes. Comencé realizando un curso de Terraform en udemy gracias a que la empresa tiene una cuenta de udemy donde puedo realizar estos cursos, gracias a esto pude realizar cursos y especializarme un poco más tanto en MongoDB como en Terraform y principalmente en Ansible que lo ocupo a diario para mi trabajo dentro de la empresa.

Para lograr relacionarme con Terraform + Azure el proceso fue crear y destruir entornos para ir encontrando fallos y entendiendo como se relacionaba cada componente de Azure entre si.

Como en la empresa trabajo constantemente con Ansible y sobre los servidores de MongoDB, no tuve que realizar tanta tarea adicional como con Terraform, pero sí hice pruebas particulares de cada componente del proyecto, independientemente unos de otros.

Hice despliegues en local de cada componente/tecnología que iba a utilizar para entender a fondo como es su infraestructura interna y comportamiento.

1. Una vez comprendido el funcionamiento de cada componente (Terraform/Azure, Ansible, Replica Set de MongoDB, y Azure Load Balancer), inicié el proyecto desde cero, comenzando a crear y relacionar cada componente para formar el proyecto completo.
2. Jenkins VM : El primer paso fue desplegar un servidor de Jenkins en una VM de azure para realizar el proceso de automatización con pipelines de Jenkins. Todo el proceso será gestionado desde esta máquina.

The screenshot shows the Jenkins web interface. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information (Lautaro Lorca). Below this is a 'Panel de Control' section. On the left sidebar, there are links for 'Nueva Tarea', 'Historial de trabajos', 'Relacion entre proyectos', 'Comprobar firma de archivos', 'Administrar Jenkins', and 'Mis vistas'. The main area displays a table of jobs with columns: 'S', 'W', 'Nombre', 'Último Éxito', 'Último Fallo', and 'Última Duración'. A job named 'FCT' is listed with a success status, last success 14 días #25, last failure 8 días 11 Hor #30, and duration 4,7 Seg. Below the table, there are filters for 'Trabajos en la cola' (No hay trabajos en la cola) and 'Estado del ejecutor de construcciones' (1 Inactivo, 2 Inactivo). At the bottom right, it says 'REST API' and 'Jenkins 2.452.1'.

2. **Creación del directorio de Terraform:** Primero, creamos el directorio de Terraform donde comenzaremos con la creación de tres grupos de recursos (resource groups) en diferentes ubicaciones de Azure para asegurarnos de que, en caso de que un servidor de Azure falle, nuestro Replica Set seguirá funcionando. Aquí no solo crearemos las maquinas virtuales sino que crearemos toda una configuración de seguridad y red para tener las VMs controladas y con una ip pública.

```

1-Terraform
├── instance-1.tf
├── main.tf
├── network.tf
├── terraform.tfstate
├── terraform.tfstate.backup
├── vars.tf
├── 2-Terraform
│   ├── instance-1.tf
│   ├── main.tf
│   ├── network.tf
│   ├── terraform.tfstate
│   ├── terraform.tfstate.backup
│   └── vars.tf
├── 3-Terraform
│   ├── instance-1.tf
│   ├── main.tf
│   ├── network.tf
│   └── terraform.tfstate
│   └── vars.tf
└── 3 directories, 17 files
    
```

Con toda la estructura creada para poder ejecutar terraform y crear la infraestructura en la nube debemos ejecutar 3 comandos:

- terraform init : Se encarga de inicializar un directorio de trabajo con archivos de configuración de Terraform, establece todos los componentes necesarios para que Terraform pueda gestionar y desplegar la infraestructura correctamente.
- terraform plan: Permite visualizar las acciones que Terraform realizará para alcanzar el estado deseado de la infraestructura, descrito en los archivos de configuración. Aquí podemos ver que al ejecutarlo tenemos 10 planes para ejecutar.

```
Plan: 10 to add, 0 to change, 0 to destroy.
```

```
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
```

```
lautaro@FCT:~/Escritorio/Tri-Terraform/1-Terraform$
```

- terraform apply: Después de generar y revisar el plan de ejecución con terraform plan, terraform apply aplica efectivamente esos cambios, creando, modificando o destruyendo recursos en el proveedor de infraestructura (como Azure).

```
ptions/c79cf5da-6a7d-4db6-b1c8-19bc84ad1997/resourceGroups/1-LLA-TFG/providers/Microsoft.KInterfaces/TFG-nic1|/subscriptions/c79cf5da-6a7d-4db6-b1c8-19bc84ad1997/resourceGroups/1-ers/Microsoft.Network/networkSecurityGroups/TFG-allow-ssh]
azurerm_virtual_machine.TFG-instance-1: Creation complete after 18s [id=/subscriptions/c796-b1c8-19bc84ad1997/resourceGroups/1-LLA-TFG/providers/Microsoft.Compute/virtualMachines/T
```

```
Apply complete! Resources: 10 added, 0 changed, 0 destroyed.
```

```
lautaro@FCT:~/Escritorio/Tri-Terraform/1-Terraform$
```

Ahora si vamos a nuestro portal de azure podremos ver la infraestructura creada, aquí veremos los 3 grupos en diferentes regiones:

Microsoft Azure

Search resources, services, and docs (G+/)

Home >

Resource groups

Conselleria d'Educació (edu.gva.es)

+ Create Manage view Refresh Export to CSV Open query Assign tags

Filter for any field... Subscription equals all Location equals all Add filter

Showing 1 to 4 of 4 records.

Name	Subscription	Location
1-LLA-TFG	Azure for Students	West Europe
2-LLA-TFG	Azure for Students	Spain Central
3-LLA-TFG	Azure for Students	Spain Central
NetworkWatcherRG	Azure for Students	West Europe

< Previous Page 1 of 1 Next >

Give feedback

Ahora si entramos dentro de cada grupo podremos ver todo lo que creamos a traves de Terraform como la VM,IP,SSH,etc.

Resource Groups:

1-LLA-TFG

1-LLA-TFG

Resource group

Search

+ Create Manage view Delete resource group Refresh Export to CSV Open query Assign tags Move Delete Export template

Essentials

Subscription (move): [Azure for Students](#) Deployments : [No deployments](#)

Subscription ID : c79cf5da-6a7d-4db6-b1c8-19bc84ad1997 Location : West Europe

Tags (edit) : [Add tags](#)

Resources

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 8 of 8 records. Show hidden types

Name	Type	Location
instance1-public-ip	Public IP address	West Europe
myosdisk1	Disk	West Europe
TFG-allow-ssh	Network security group	West Europe
TFG-network	Virtual network	West Europe
TFG-nic	Network Interface	West Europe
TFG-nic1	Network Interface	West Europe
TFG-public-ip	Public IP address	West Europe
TFG-vm1	Virtual machine	West Europe

2-LLA-TFG

« 2-LLA-TFG Resource group »

Search «

+ Create Manage view Delete resource group Refresh Export to CSV Open query Assign tags Move Delete Export template

Overview

Activity log

Access control (IAM)

Tags

Resource visualizer

Events

Settings

Deployments

Security

Deployment stacks

Policies

Properties

Locks

Cost Management

Cost analysis

Cost alerts (preview)

Budgets

Advisor recommendations

Essentials

Subscription (move): [Azure for Students](#)

Subscriptions ID: c79cf5da-6a7d-4db6-b1c8-19bc84ad1997

Deployments: [No deployments](#)

Location: Spain Central

Tags (edit): [Add tags](#)

Resources Recommendations

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 6 of 6 records. Show hidden types No grouping

Name	Type	Location
myosdisk1	Disk	Spain Central
TFG-2-allow-ssh	Network security group	Spain Central
TFG-2-network	Virtual network	Spain Central
TFG-2-nic-2	Network Interface	Spain Central
TFG-2-public-ip-2	Public IP address	Spain Central
TFG-2-vm2	Virtual machine	Spain Central

3-LLA-TFG

« 3-LLA-TFG Resource group »

Search «

+ Create Manage view Delete resource group Refresh Export to CSV Open query Assign tags Move Delete Export template

Overview

Activity log

Access control (IAM)

Tags

Resource visualizer

Events

Settings

Deployments

Security

Deployment stacks

Policies

Properties

Locks

Cost Management

Cost analysis

Cost alerts (preview)

Budgets

Advisor recommendations

Essentials

Subscription (move): [Azure for Students](#)

Subscriptions ID: c79cf5da-6a7d-4db6-b1c8-19bc84ad1997

Deployments: [No deployments](#)

Location: Spain Central

Tags (edit): [Add tags](#)

Resources Recommendations

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 6 of 6 records. Show hidden types No grouping

Name	Type	Location
myosdisk1	Disk	Spain Central
TFG-3-allow-ssh	Network security group	Spain Central
TFG-3-network	Virtual network	Spain Central
TFG-3-nic-3	Network Interface	Spain Central
TFG-3-public-ip-3	Public IP address	Spain Central
TFG-3-vm3	Virtual machine	Spain Central

Y podemos comprobar el acceso a las 3 vms que lo tenes permitido por clave ssh desde el inventory:

The image contains three terminal screenshots. The leftmost screenshot shows the output of 'apt list' and 'sudo pro status', indicating that 12 updates can be applied immediately. The middle screenshot shows the output of 'whoami' (lautaro) and 'ls' (listing files in /usr/share/doc/*), and the output of 'sudo apt update' (showing updates are more than a week old). The rightmost screenshot shows the output of 'whoami' (lautaro) and 'ls' (listing files in /usr/share/doc/*).

3. **Desarrollo de los playbooks de Ansible:** Una vez configuradas las máquinas, comenzamos con el desarrollo de los playbooks de Ansible. Estos playbooks se encargan de la instalación del Replica Set de MongoDB y también incluyen scripts para actualizar MongoDB según salgan nuevas versiones. La estructura de ansible se basa en 2 directorios (ansible,inventory), en ansible solo se encontraran los playbooks que son los encargados de ejecutar cualquier tarea y luego el inventory que es donde estan todas las especificaciones de los host/servidores aquí se encontrarán todas las variables particulares de cada host, el nombre, la conexión y todo lo necesario de cada host.

The image shows a terminal screenshot with the following commands and output:
 • `lautaro@FCT:~/Escritorio/Tri-Terraform/Ansible$ ls ansible/`
 db.yml roles utils
 • `lautaro@FCT:~/Escritorio/Tri-Terraform/Ansible$ ls inventory/lab/`
 group_vars hosts host_vars role_parents
 • `lautaro@FCT:~/Escritorio/Tri-Terraform/Ansible$`

4. **Configuración del Load Balancer:** Configuramos el balanceador de carga (Load Balancer) con nginx para que maneje el tráfico hacia los servidores y distribuya las peticiones de manera eficiente, asegurando que las consultas (queries) se dirijan correctamente a los servidores disponibles.
5. **Monitoreo con Prometheus y Grafana:** Configuramos Prometheus y Grafana en la máquina de Jenkins para monitorear los servidores. Esto nos permitirá tener una visión clara y en tiempo real del estado y rendimiento de nuestros servicios, facilitando la detección y resolución de posibles problemas. Crearemos nuestros propios dashboards y plantillas para lograr tener una monitorización buena de nuestro cluster.

En resumen, la planificación del proyecto implica una serie de pasos bien definidos para asegurar la automatización y escalabilidad de la infraestructura en la nube, garantizando alta disponibilidad y eficiencia operativa. Cada componente ha sido cuidadosamente integrado y probado para cumplir con los objetivos del proyecto y proporcionar un entorno robusto y fiable.

CONCLUSIONES

Personalmente, al reflexionar sobre este proyecto y mi experiencia en Lorient durante estos meses, así como mis dos años cursando el grado superior de ASIR, puedo decir que el nivel de aprendizaje y desarrollo personal que he alcanzado ha sido sorprendente. Trabajar en Lorient y sumergirme en el mundo laboral ha abierto un amplio abanico de oportunidades de aprendizaje.

Ingresé al ciclo superior sin ningún conocimiento previo en el ámbito informático, y hoy estoy desarrollando proyectos como este. Esto es algo que no habría imaginado al comenzar el curso. Los dos años de estudio y la experiencia en Lorient me han dado las herramientas y el conocimiento necesario para desarrollar proyectos complejos como este, utilizando tecnologías de vanguardia y prácticas de automatización.

En cuanto a las conclusiones específicas del proyecto, creo que realmente ha cambiado mi enfoque frente a cualquier situación en la vida. Ahora entiendo la importancia de pensar en soluciones que no solo resuelvan problemas momentáneos, sino que también estén diseñadas para ser escalables y sostenibles a largo plazo. Esto es fundamental tanto en el entorno empresarial como en la vida personal.

Este proyecto se centra en la automatización no solo por el hecho de automatizar acciones o procesos, sino porque entiendo que es crucial para el crecimiento de una empresa, la eficiencia en el tiempo de los empleados y para realizar un trabajo que no solo esté bien hecho, sino que también ahorre recursos críticos para el funcionamiento diario de una empresa.

En resumen, este proyecto no solo me ha permitido aplicar mis habilidades técnicas y conocimientos adquiridos, sino que también me ha enseñado lecciones valiosas sobre la importancia de la automatización, la planificación a largo plazo y el crecimiento personal y profesional.

REFERENCIAS

Documentación utilizada:

Ansible - <https://www.ansible.com/>

Terraform - <https://www.terraform.io/>

Mongodb - <https://www.mongodb.com/docs/>

Microsoft Azure - <https://learn.microsoft.com/en-us/azure/?product=popular>

Terraform + Azure - <https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs>

Udemy - https://www.udemy.com/?utm_source=aff-campaign&utm_medium=udemyads&LSNPUBID=OzpaRYwFVro&ranMID=47901&ranEAID=OzpaRYwFVro&ranSiteID=OzpaRYwFVro-q3b9xb7OwzWMkSNenJyMcg

Load Balancer - <https://azure.microsoft.com/es-es/products/load-balancer/>

Jenkins - <https://www.jenkins.io/doc/>

Gitlab - <https://docs.gitlab.com/>

Genially -

<https://app.genially.com/teams/6668a2418fecf10013d0066a/spaces/6668a2418fecf10013d00687/dashboar>