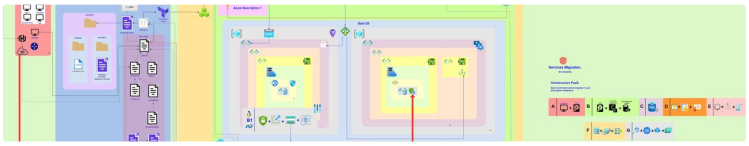


Azure: Diagrama de servicios Digicem!

Un saludo, espero se encuentre bien.

Tiempo de lectura: 3 minutos.

Description.

A continuación detallaremos, desde **Infraestructura Digicem - Cloud**, los documentos reflejados como:

- Mapa actualizado de los servicios desplegados bajo IaC, como migración de la Infraestructura legada y deployada en el :azure: portal.
- Distribución de repositorios y entornos dentro de la compañía.
- Ciclo de vida de servicios DevOps en la subida de un Pull Request (ga) o release con un paso a produccion bajo un Merge Request (rc).

Walkthrougth.

- :terraform:: Herramienta de administración de entornos y despliegues de recursos y servicios dentro del :azure: portal.
- :azure: **Repos:** Herramienta de gestión de pipelines en pasos por el branching strategy declarado de git-flow y trunk, entre los entornos: Dev, STA y Prod, algunos HotFixes y Features de recuperación de aplicaciones comprometidas.
- :azure: **Pipelines:** Herramienta de gestión de pipelines en cada una de las fases DevOps según stages declarados.

Stages.

Requerimientos.

- Tener una :azure: suscripción.
- Tener un equipo local donde administrar cambios de repositorios y subirlos hacia los repositorios :terraform: , JSON en :azure: DevOps, sobre Repos para con los manifiestos (Terraform, :azure: CLI).
- Crear o clonar un proyecto con repositorios y modulos :terraform:, usando la base de estructura a estandarizar.

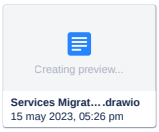
Emojis 👇

:azure: **Plataforma del Portal de Azure - PaaS.**

:terraform: **Terraform.**

Publicaciones.

- **Diagrama de servicios.**
- **Proyecto.**



Services Migrat... .drawio
15 may 2023, 05:26 pm

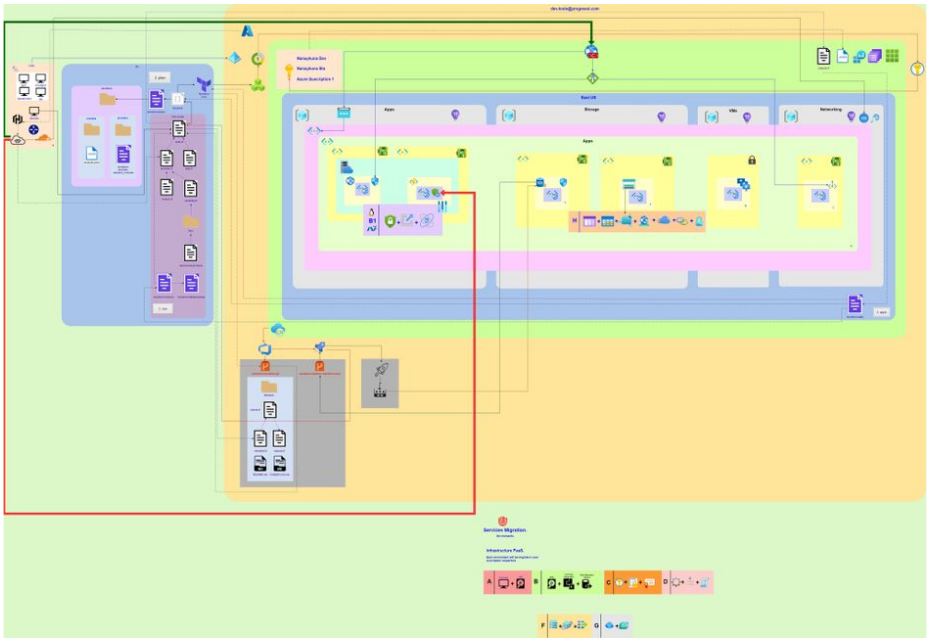


Diagrama de servicios Nanykuna en :azure:**.**

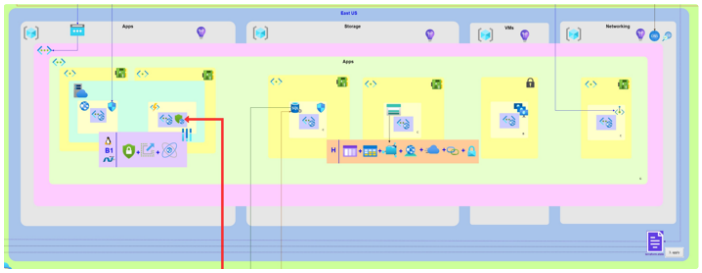
:azure: Cloud conceptos.

Construcción.

- Por favor, puedes hacer uso de los siguientes diagramas para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

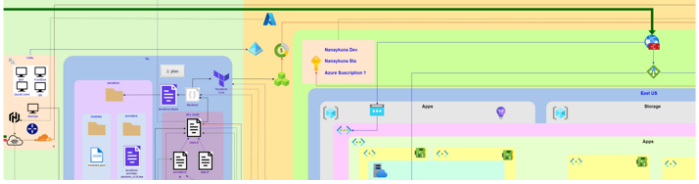
A continuación se muestran los conceptos / terminologías centrales que se utilizan en :azure: :

- **Nomenclatura de recursos**: Administramos los recursos a través de nombrarlos con una notación o nemotecnia que referencie el tipo de servicio, organización, ambiente, herramienta de despliegue, zona de ubicación y el nombre del recurso, separados por un guion a excepción de los recursos que no aceptan este carácter especial (-) dentro de su objeto.
- **Grupo de recursos**: Administramos los recursos a través de cuatro resources groups, que segmentan según nivel de servicio.



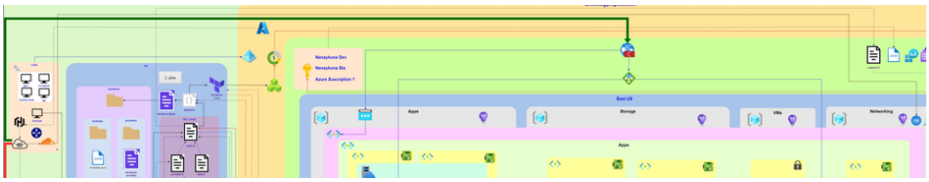
- **Resource Groups.**

- **WAF**: Desde Internet se determinan reglas inbound u outbound hacia direcciones ip públicas, zonas geográficas.



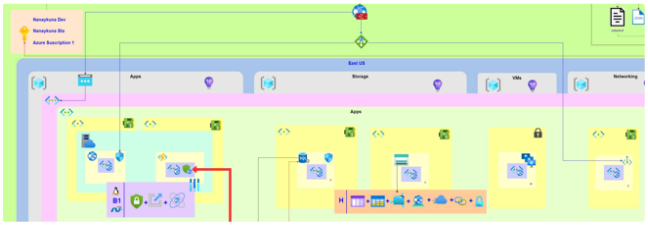
- **WAF Rules.**

- **DNS**: Desde Internet se redirecciona la petición u traducción de la dirección ip desde dominios web, administrado por CloudFlare, hacia los servicios internos LAN por cada zona privada.



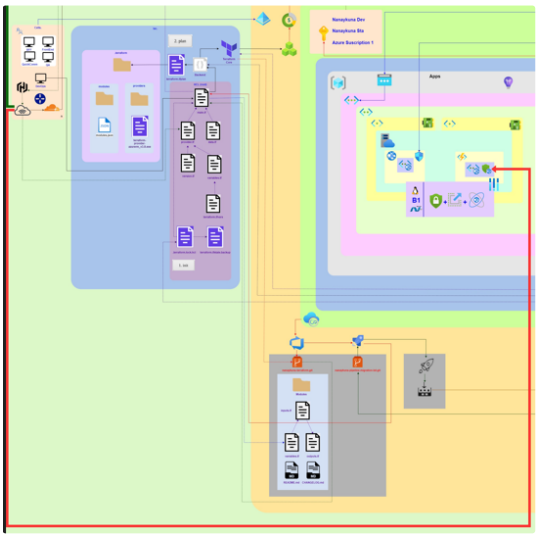
- **Private Zones DNS.**

- **Application Gateway**: Balancea las peticiones hacia cada App.



- **Load Balancing.**

- **Function Apps**: Desde Internet se bloquea el acceso hacia los endpoints (Private Links) gestionados de cada function privadas de nuestra Virtual Net, accediendo solo por medio de nuestras Web Apps públicas.



- **Access Public Blocks.**

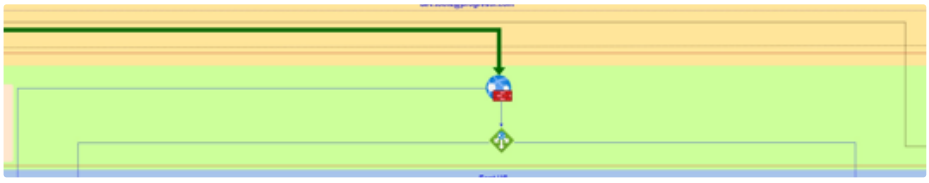
Despliegues.

> - Por favor, puedes hacer uso de la información para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

:azure: Ciclo de vida.

A continuación se muestran los conceptos / terminologías centrales que se utilizan en el ciclo de vida de :azure: :

- **Suscripciones:** Administramos los recursos a través de un git Flow como estrategia de branching, desde ambientes locales bajos hacia el paso de release en cada sprint (Scrum), de quince días llegando a freezing de pull requests hacia cada merge según version y tagging en release.



Suscriptions.

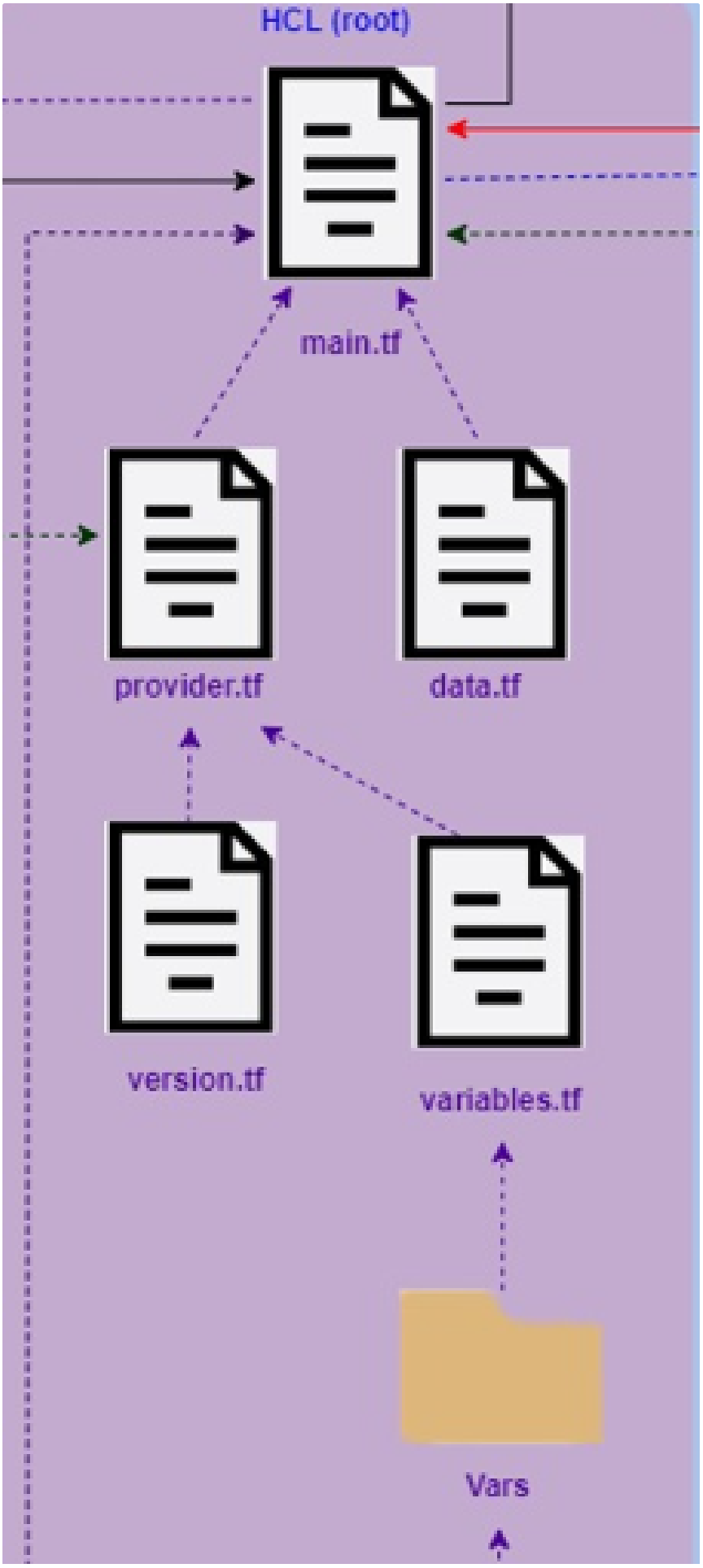
:terraform: IaC conceptos.

Construcción.

- Por favor, puedes hacer uso de los siguientes diagramas para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

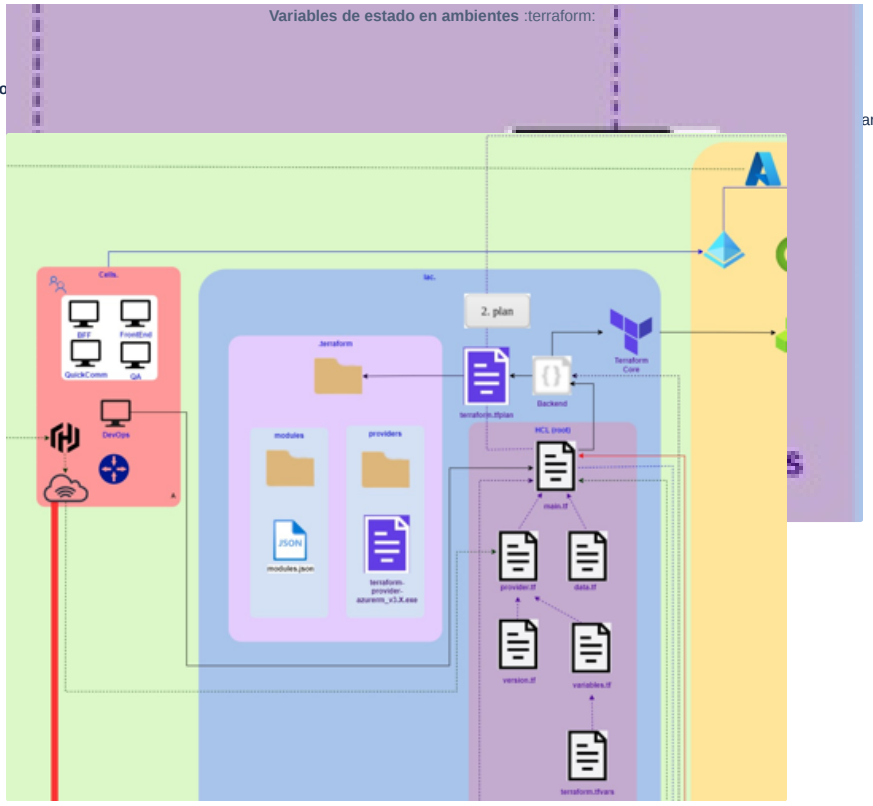
A continuación se muestran los conceptos / terminologías centrales que se utilizan en :terraform::

- **IaC**: Fase del ciclo de vida DevOps, como despliegue de infraestructura como código. Con ello evitar errores humanos a la hora de actualizar la infraestructura, aumentar la seguridad.
- **Variables**: También se utiliza como variables de entrada, es un par clave-valor utilizado por los módulos Terraform para permitir la personalización.
  - Nosotros hacemos uso de una estructura de variables dentro del fichero local **"Vars"** que almacena en el archivo :terraform: "**terraform.tfvars**" para cada ambiente, todos los valores generales de acceso, como redes, direcciones ip, CIDR, etiquetas, ambientes de desarrollo: Dev, Sta, Prod.
  - Donde… Desde el archivo :terraform: "**variables.tf**" consumiremos dichos valores, bajo la interpolaridad de :terraform:, como el escalamiento de variables globales y de estado.
  - Estas variables serán llamadas en el archivo :terraform: "**main.tf**" para consumo en los modulos y recursos locales requeridos.


HCL (root)
main.tf
provider.tf
data.tf
version.tf
variables.tf
Vars

- **Variables de estado en ambientes** :terraform:

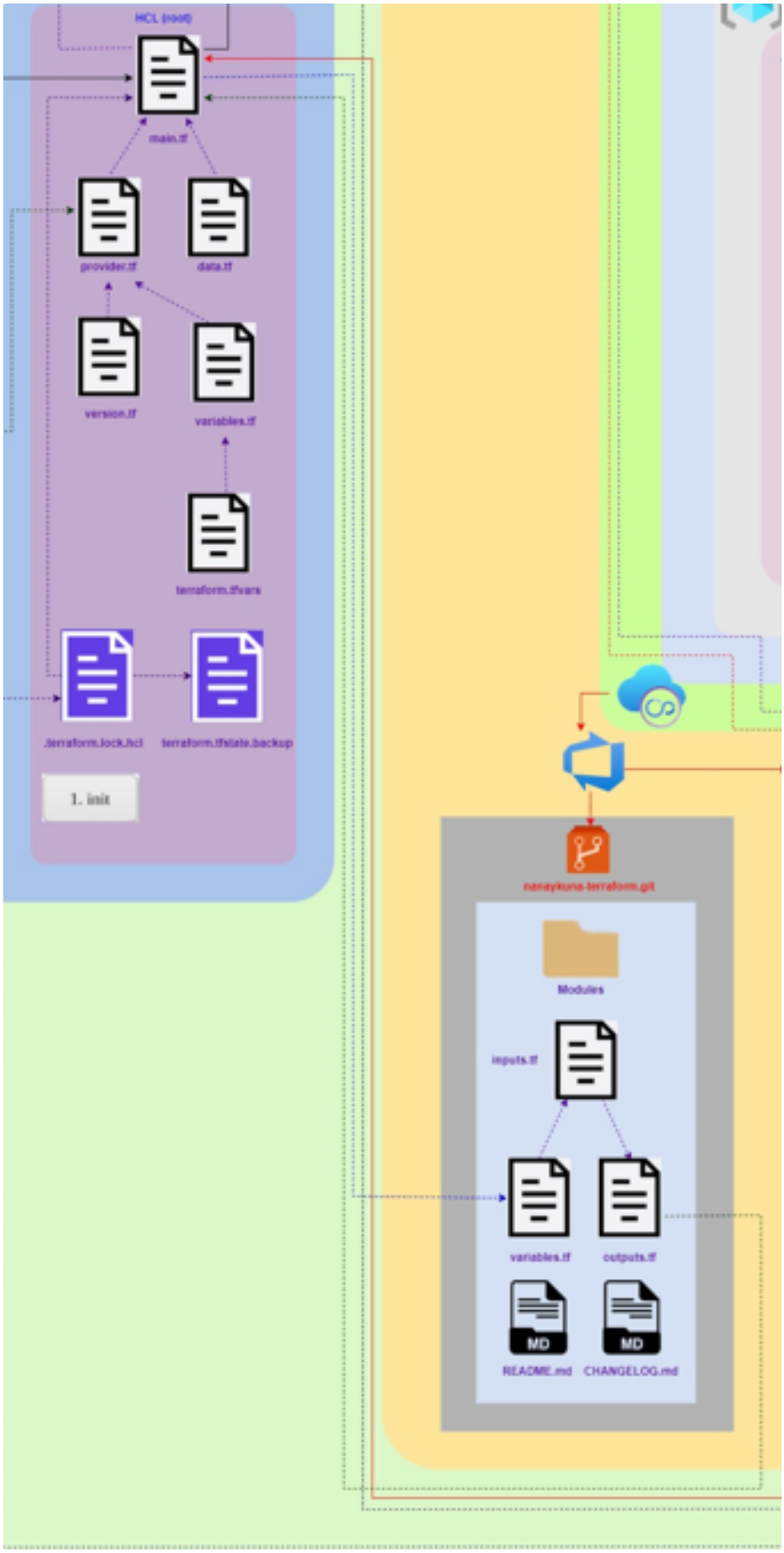
- **Pro**
  - argumentos, sintaxis.



Descarga de data desde el proveedor azure, hacia hashicorp.

  - Donde… Desde el archivo :terraform: "**version.tf**" consumiremos la version de uso corriendo en :terraform:.
  - Estas variables serán llamadas en el archivo :terraform: "**provider.tf**" para consumo en los modulos y recursos locales requeridos.
  - Al iniciar el paso de recursos, se crea en el proyecto local un directorio oculto :terraform: llamado "**.terraform**".
  - El anterior contiene dos subdirectorios, uno para modulos con formatos JSON y el otro donde se descarga el executable del provider.
  - Dentro de las features del provider, se deben establecer los argumentos requeridos para proteger servicios y recursos ante una mala manipulación de terceros.

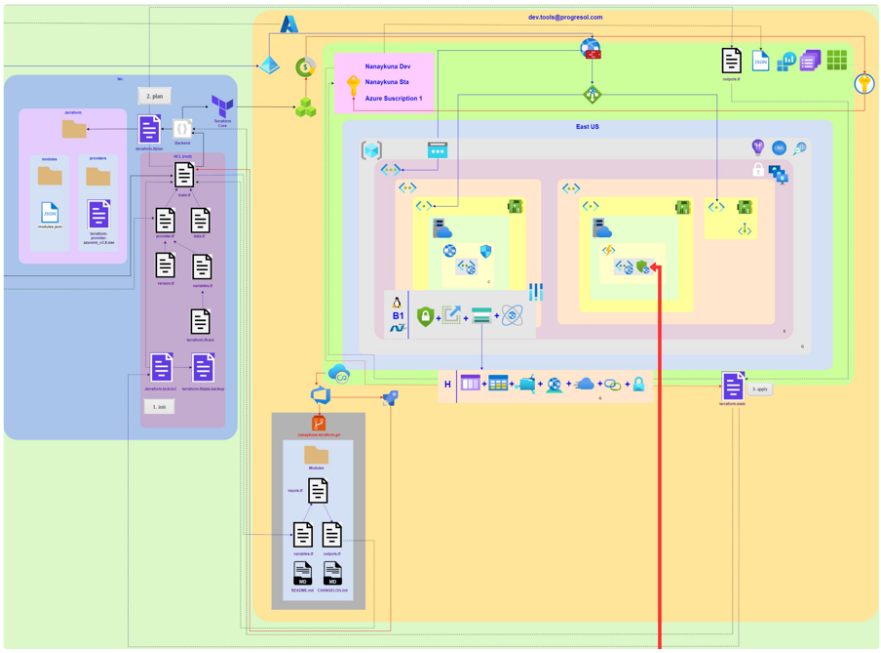
- **Módulo**: Es una carpeta con plantillas Terraform donde se definen todas las configuraciones.
  - Nosotros hacemos uso de modulos customizados a nuestros recursos, servicios y continuidad del negocio.
  - Los cuales son almacenados remotamente con su respectiva estructura de archivos de variables, outputs y main.



llamada de modulos desde el repositorio remoto.

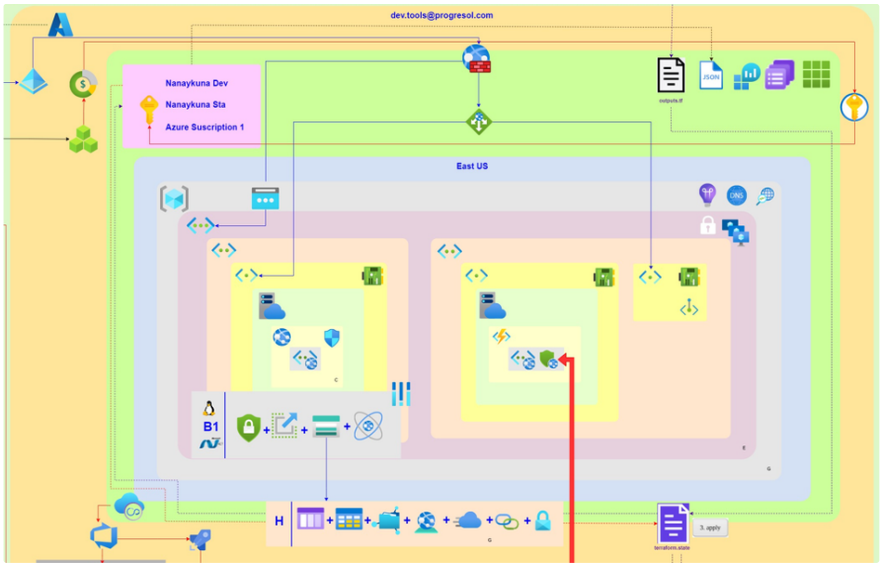
- El archivo :terraform: "**variables**.tf" recibirá todos los inputs del archivo :terraform: "**main.tf"** local.
- Estas variables serán presentadas en el archivo :terraform: "**main.tf"** remoto para la construcción en los modulos, según los recursos locales requeridos.
- Obteniendo salidas de configuraciones de archivos :terraform: "**outputs.tf**".
- Estas salidas serán llamadas en el archivo :terraform: "**main.tf**" local para consumo en los nuevos modulos y recursos locales requeridos.

- **Estado**: Consiste en información en caché sobre la infraestructura administrada por Terraform y las configuraciones relacionadas.
  - Al finalizar el paso de recursos, se crea en el proyecto local un archivo oculto :terraform: llamado "**.terraform.lock.hcl**".
  - El anterior contiene bloquea el paso de despliegues, desde dos o más cambios en simultaneo.
  - Se debe almacenar el mismo en un:key-vault:con el fin de centralizar el recurso.
  - Luego de estar libre el mismo, se guarda una copia de respaldo en un archivo :terraform: "**terraform.tfstate.backup**".
  - El anterior es el resultado local, que se debe centralizar. El cual se ejecuta al finalizar un plan, luego el apply crearía el archivo de estado.



- **Uso del estado de** :terraform: **para almacenar los cambios en los recursos y servicios.**

- **Recursos**: Se refiere a un bloque de uno o más objetos de infraestructura (instancias de cómputo, redes virtuales, etc.), que se utilizan para configurar y administrar la infraestructura.
  - Cada recurso deployado, sigue un estándar en nomenclatura para fácil identificación en el servicio consumido.



- **Recursos deployados en** :azure:**.**

Servicios deployados o activados en :azure:**.**

- **Fuente de datos**: Los proveedores lo implementan para devolver información sobre objetos externos a terraform.

Pruebas.

- Por favor, puedes hacer uso de la información para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

- **Valores de salida**: Estos son valores de retorno de un módulo terraform que pueden ser utilizados por otras configuraciones.
  - Se establecen los valores de salidas customizados, para validar el nombramiento de los recursos y servicios, igual la protección de texto plano ante data sensible.

Despliegues.

- Por favor, puedes hacer uso de la información para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

:terraform: Ciclo de vida.

A continuación se muestran los conceptos / terminologías centrales que se utilizan en el ciclo de vida de :terraform::

- **Planificación**: Es una de las etapas en las que determina qué se debe crear, actualizar o destruir para pasar del estado real / actual de la infraestructura al estado deseado.

- **Aplicar**: Es una de las etapas donde se aplican los cambios de estado real / actual de la infraestructura para pasar al estado deseado.

Emojis 👇

:azure: **Plataforma del Portal de Azure - PaaS.**

:terraform: **Terraform.**

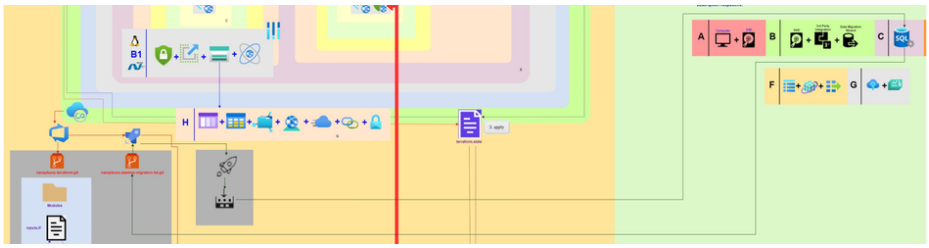
:azure: CICD conceptos.

Construcción.

- Por favor, puedes hacer uso de los siguientes diagramas para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

A continuación se muestran los conceptos / terminologías centrales que se utilizan en :azure: :

CICD: Fase del ciclo de vida DevOps, como herramienta donde se construye un valor agregado en la actualización de uno o varios pasos a produccion controlados, como versionados de manera más optima, efectiva y segura, dentro de la estrategia de branching que utilizamos en la compañía.



- **Flujo CI CD.**

Pruebas.

- Por favor, puedes hacer uso de la información para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

- **Artifacts**: Puedo contener la ruta del repositorio, como elementos a presentar al pipeline o release para desencadenar un test ya sea unitario, calidad QA, Seguridad SAST (vulnerabilidades) y performance HPC (estrés), dentro la integración continua.

Despliegues.

- Por favor, puedes hacer uso de la información para uso libre o modificación del mismo, respetando el versionamiento de esta página en Confluence.

:azure: Ciclo de vida.

- **Release**: Trigger para la entrega continua.

Emojis 👇

:azure: **Plataforma del Portal de Azure - PaaS.**

💪Ahora se puede dimensionar el nivel de ejecución, como tiempo a la hora de gestionar este requerimiento.

¡Mil gracias por la atención prestada!

Cualquier duda me puedes contactar…

:WhatsApp: ***+573058288031***

como mi usuario :slack: