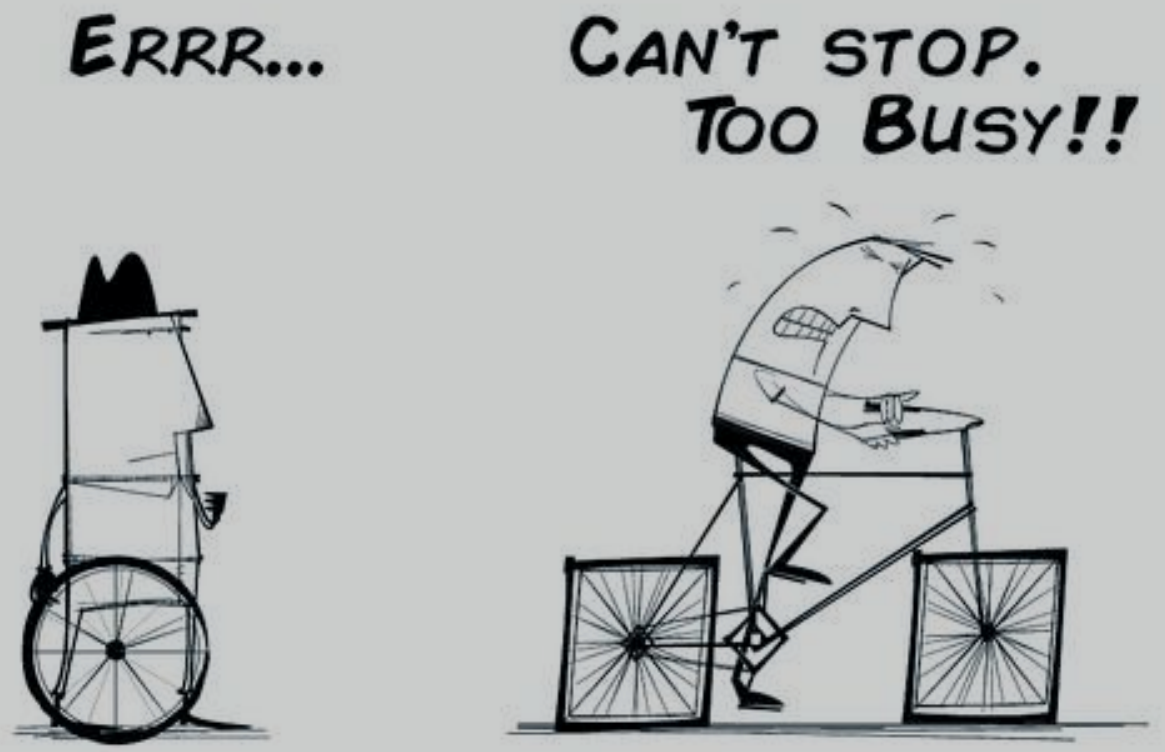
ASYS

Por una humanidad sustentable

Dev Ops

Modulo 7



TOO BUSY TO IMPROVE?

WorkCompass

MAPA

DEMO

PRESENTACIÓN

REPETIR DEMO

Objetivos del Modulo

1. Repaso
2. Hilo Rojo
3. Terraform
4. Kahoot

Qué es DevOps ?

Desarrollar, Operar y Mejorar Sistemas de Excelente Calidad

Flujo

Hacer el trabajo

Lo más eficiente posible

Retro Alimentación

Observar
Medir
Notificar

Mejora Continua

Incrementar el valor de nuestro trabajo

Cultura

Flujo

Intro

Trabajo

Diseño

Calidad

Infraestructura

FrontEnd

BackEnd

DataEnd

Infraestructura de Producción on Demand

Pipeline - C.I. - C.D. - C.Deployment

Cultura

Retroalimentación

- Telemetría FrontEnd
- Telemetría BackEnd
- Telemetría Negocio
- Telemetría Infraestructura

Mejora Continua

- Operaciones
- Experimentación
- Ingeniería del Caos

Infraestructura como código

Será posible ser uno feliz administrando infraestructura ?

Terraform

Herramienta

Para la administración de Recursos de infraestructura tales como (por nombrar solo algunos)

- Computadoras físicas
- Computadoras virtuales
- Redes
- DNS
- CDNs
- Administradores de Contenedores
- Balanceadores de carga

Providers

Son proveedores de recursos tales como

- AWS
- AZURE
- GCP
- OpenStack
- Heroku
- DigitalOcean
- CloudFlare
- DNSimple

Estos hacen módulos y los ponen en el registro de terraform. Estos módulos son los responsables de interactuar con las API´s del proveedor y exponer sus recursos

Como instalar Terraform

?

Instalar

1 Descargar binario

2 Ubicar el archivo en un directorio adecuado

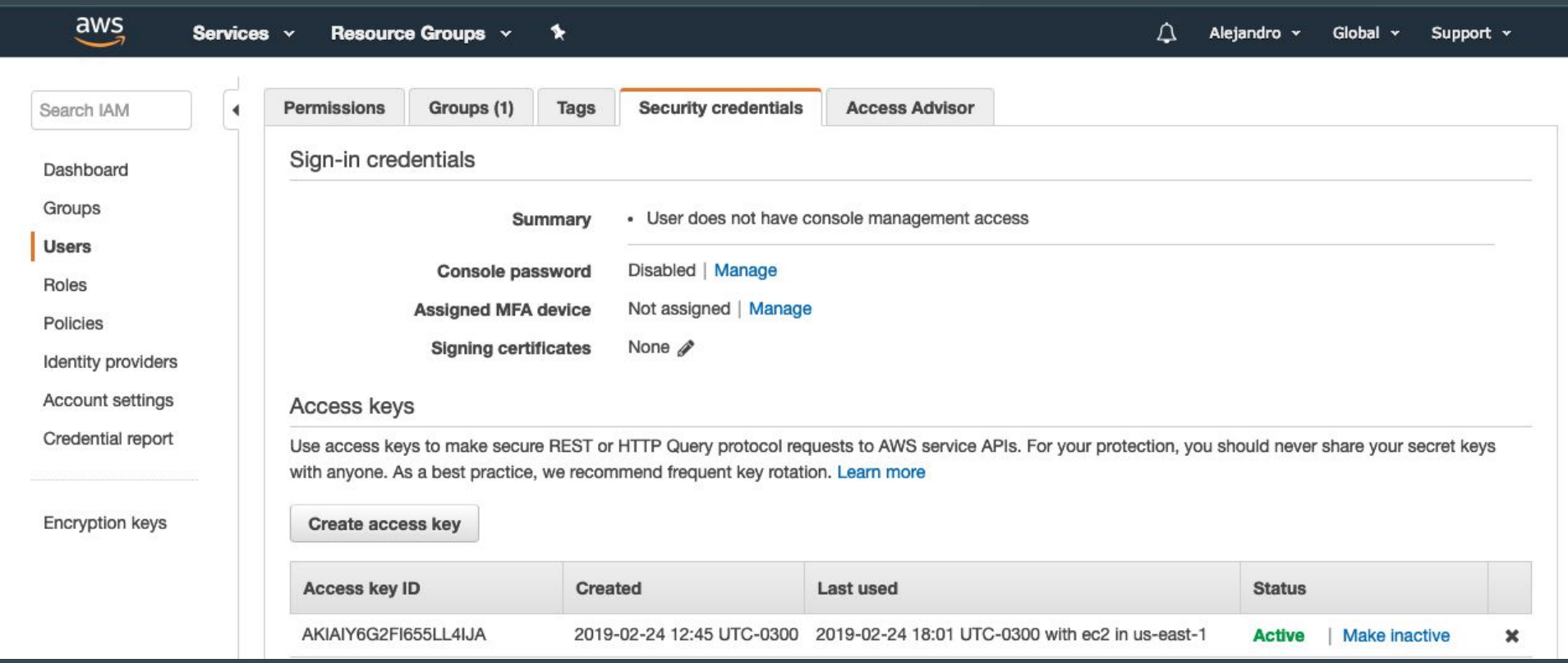
3 Si hace falta : agregar el directorio al $PATH del sistema operativo

```
 abernal@Alejandros-MacBook-Pro  ~  which terraform
/Users/abernal/bin/terraform
 abernal@Alejandros-MacBook-Pro  ~  echo $PATH
/Library/Frameworks/Python.framework/Versions/3.6/bin:/Users/abernal/bin:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/l
t:~/.dotnet/tools:/Library/Frameworks/Mono.framework/Versions/Current/Commands:/Applications/Wireshark.app/Contents/MacOS
 abernal@Alejandros-MacBook-Pro  ~  terraform
Usage: terraform [-version] [-help] <command> [args]

The available commands for execution are listed below.
The most common, useful commands are shown first, followed by
less common or more advanced commands. If you're just getting
started with Terraform, stick with the common commands. For the
other commands, please read the help and docs before usage.

Common commands:
    apply              Builds or changes infrastructure
    console            Interactive console for Terraform interpolations
    destroy            Destroy Terraform-managed infrastructure
    env                Workspace management
    fmt                Rewrites config files to canonical format
    get                Download and install modules for the configuration
    graph              Create a visual graph of Terraform resources
    import             Import existing infrastructure into Terraform
    init               Initialize a Terraform working directory
    output             Read an output from a state file
    plan               Generate and show an execution plan
    providers          Prints a tree of the providers used in the configuration
    push               Upload this Terraform module to Atlas to run
    refresh            Update local state file against real resources
    show               Inspect Terraform state or plan
    taint              Manually mark a resource for recreation
    untaint            Manually unmark a resource as tainted
    validate           Validates the Terraform files
    version            Prints the Terraform version
    workspace          Workspace management

All other commands:
```

Como crear una VM en AWS

?

1 Obtener las credenciales en AWS para interactuar con las API´s



2 Crear un archivo con extensión *.tf

```
provider "aws" {
  access_key = "ACCESS_KEY_HERE"
  secret_key = "SECRET_KEY_HERE"
  region     = "us-east-1"
}

resource "aws_instance" "example" {
  ami           = "ami-2757f631"
  instance_type = "t2.micro"
}
```

3 Ejecutar el comando terraform init

```
abernal@Alejandros-MacBook-Pro ~/tmp/terraform-ppt terraform init

Initializing provider plugins...
- Checking for available provider plugins on https://releases.hashicorp.com...
- Downloading plugin for provider "aws" (1.60.0)...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.aws: version = "~> 1.60"

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

4 Ejecutar el comando terraform plan

```
abernal@Alejandros-MacBook-Pro ~/tmp/terraform-ppt terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.


------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  + aws_instance.example
      id:                           <computed>
      ami:                          "ami-2757f631"
```

5 Ejecutar el comando terraform apply

```
abernal@Alejandros-MacBook-Pro ~/tmp/terraform-ppt terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  + aws_instance.example
      id:                               <computed>
      ami:                              "ami-2757f631"
      arn:                              <computed>
      associate_public_ip_address:      <computed>
      availability_zone:                <computed>
      cpu_core_count:                   <computed>
      cpu_threads_per_core:             <computed>
      ebs_block_device.#:               <computed>
      ephemeral_block_device.#:         <computed>
      get_password_data:                "false"
      host_id:                          <computed>
      instance_state:                   <computed>
      instance_type:                    "t2.micro"
      ipv6_address_count:               <computed>
      ipv6_addresses.#:                 <computed>
      key_name:                         <computed>
      network_interface.#:              <computed>
      network_interface_id:             <computed>
      password_data:                    <computed>
      placement_group:                  <computed>
      primary_network_interface_id:     <computed>
      private_dns:                      <computed>
      private_ip:                       <computed>
      public_dns:                       <computed>
      public_ip:                        <computed>
      root_block_device.#:              <computed>
      security_groups.#:                <computed>
      source_dest_check:                "true"
      subnet_id:                        <computed>
      tenancy:                          <computed>
      volume_tags.%:                    <computed>
      vpc_security_group_ids.#:         <computed>


Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value:
```

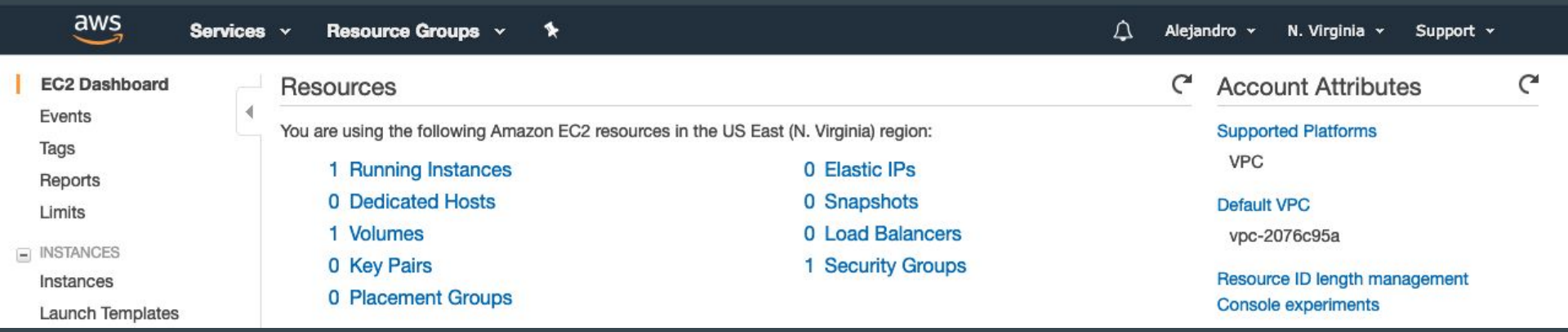
6 Confirmar ejecución

```
abernal@Alejandros-MacBook-Pro ~/tmp/terraform-ppt terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  + aws_instance.example
      id:                               <computed>
      ami:                              "ami-2757f631"
      arn:                              <computed>
      associate_public_ip_address:      <computed>
      availability_zone:                <computed>
      cpu_core_count:                   <computed>
      cpu_threads_per_core:             <computed>
      ebs_block_device.#:               <computed>
      ephemeral_block_device.#:         <computed>
      get_password_data:                "false"
      host_id:                          <computed>
      instance_state:                   <computed>
      instance_type:                    "t2.micro"
      ipv6_address_count:               <computed>
      ipv6_addresses.#:                 <computed>
      key_name:                         <computed>
      network_interface.#:              <computed>
      network_interface_id:             <computed>
      password_data:                    <computed>
      placement_group:                  <computed>
      primary_network_interface_id:     <computed>
      private_dns:                      <computed>
      private_ip:                       <computed>
      public_dns:                       <computed>
      public_ip:                        <computed>
      root_block_device.#:              <computed>
      security_groups.#:                <computed>
      source_dest_check:                "true"
      subnet_id:                        <computed>
      tenancy:                          <computed>
      volume_tags.%:                    <computed>
      vpc_security_group_ids.#:         <computed>


Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes
```

7 Verificando

```
aws_instance.example: Creating...
  ami:                               "" => "ami-2757f631"
  arn:                               "" => "<computed>"
  associate_public_ip_address:       "" => "<computed>"
  availability_zone:                 "" => "<computed>"
  cpu_core_count:                    "" => "<computed>"
  cpu_threads_per_core:              "" => "<computed>"
  ebs_block_device.#:                "" => "<computed>"
  ephemeral_block_device.#:          "" => "<computed>"
  get_password_data:                 "" => "false"
  host_id:                           "" => "<computed>"
  instance_state:                    "" => "<computed>"
  instance_type:                     "" => "t2.micro"
  ipv6_address_count:                "" => "<computed>"
  ipv6_addresses.#:                  "" => "<computed>"
  key_name:                          "" => "<computed>"
  network_interface.#:               "" => "<computed>"
  network_interface_id:              "" => "<computed>"
  password_data:                     "" => "<computed>"
  placement_group:                   "" => "<computed>"
  primary_network_interface_id:      "" => "<computed>"
  private_dns:                       "" => "<computed>"
  private_ip:                        "" => "<computed>"
  public_dns:                        "" => "<computed>"
  public_ip:                         "" => "<computed>"
  root_block_device.#:               "" => "<computed>"
  security_groups.#:                 "" => "<computed>"
  source_dest_check:                 "" => "true"
  subnet_id:                         "" => "<computed>"
  tenancy:                           "" => "<computed>"
  volume_tags.%:                     "" => "<computed>"
  vpc_security_group_ids.#:          "" => "<computed>"
aws_instance.example: Still creating... (10s elapsed)
aws_instance.example: Still creating... (20s elapsed)
aws_instance.example: Still creating... (30s elapsed)
aws_instance.example: Still creating... (40s elapsed)
aws_instance.example: Creation complete after 41s (ID: i-0446dbfa55e20bd92)

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

7 Verificando



Como eliminar nuestra VM en AWS

?

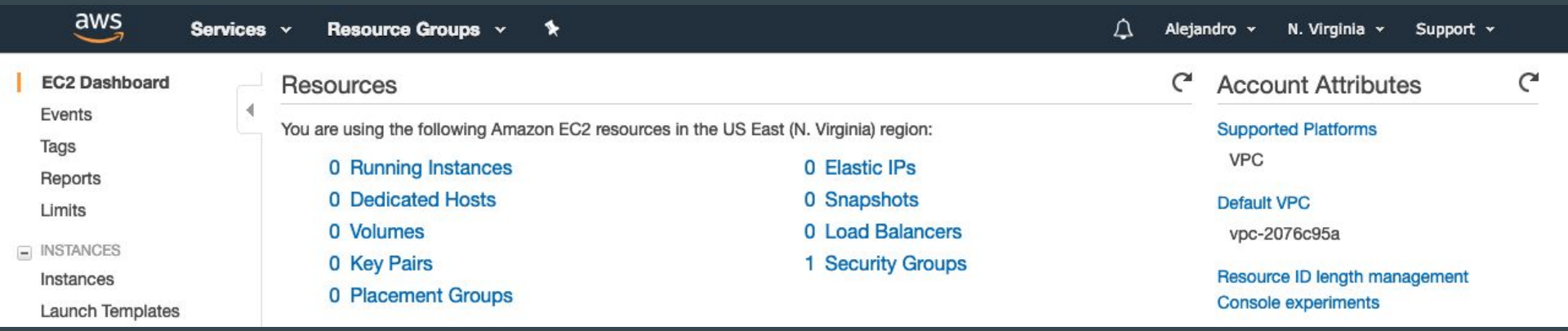
1 Ejecutar el comando terraform destroy

```
abernal@Alejandros-MacBook-Pro ~/tmp/terraform-ppt terraform destroy
aws_instance.example: Refreshing state... (ID: i-0446dbfa55e20bd92)

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  - aws_instance.example


Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes
```

1 Ejecutar el comando terraform destroy

```
aws_instance.example: Destroying... (ID: i-0446dbfa55e20bd92)
aws_instance.example: Still destroying... (ID: i-0446dbfa55e20bd92, 10s elapsed)
aws_instance.example: Still destroying... (ID: i-0446dbfa55e20bd92, 20s elapsed)
aws_instance.example: Still destroying... (ID: i-0446dbfa55e20bd92, 30s elapsed)
aws_instance.example: Still destroying... (ID: i-0446dbfa55e20bd92, 40s elapsed)
aws_instance.example: Still destroying... (ID: i-0446dbfa55e20bd92, 50s elapsed)
aws_instance.example: Still destroying... (ID: i-0446dbfa55e20bd92, 1m0s elapsed)
aws_instance.example: Destruction complete after 1m6s

Destroy complete! Resources: 1 destroyed.
```

2 Verificando



RESUMEN

PARA QUÉ SIRVE

terraform init

terraform plan

terraform apply

terraform show

terraform show

TERRAFORM WEB SITE

LIBRO

PARROQUIALES