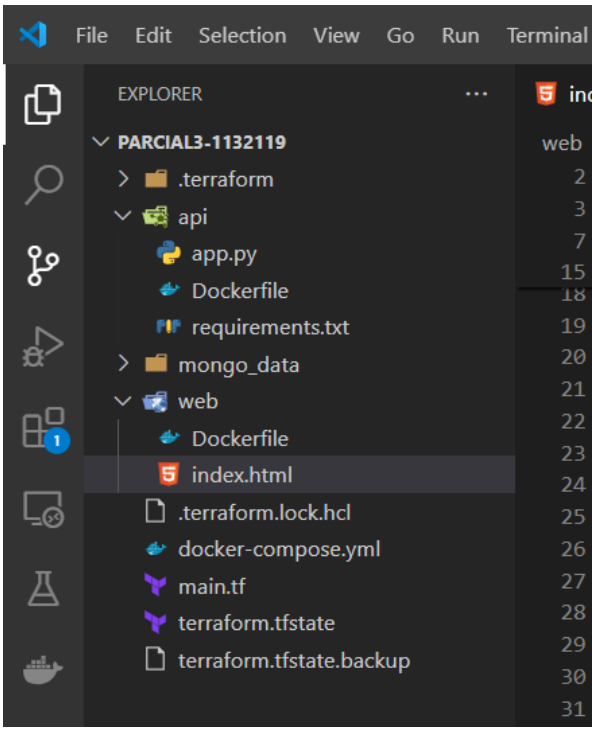
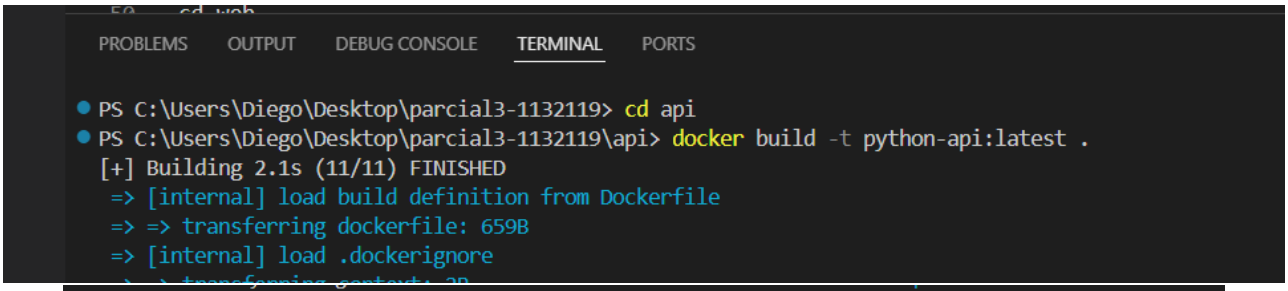
Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

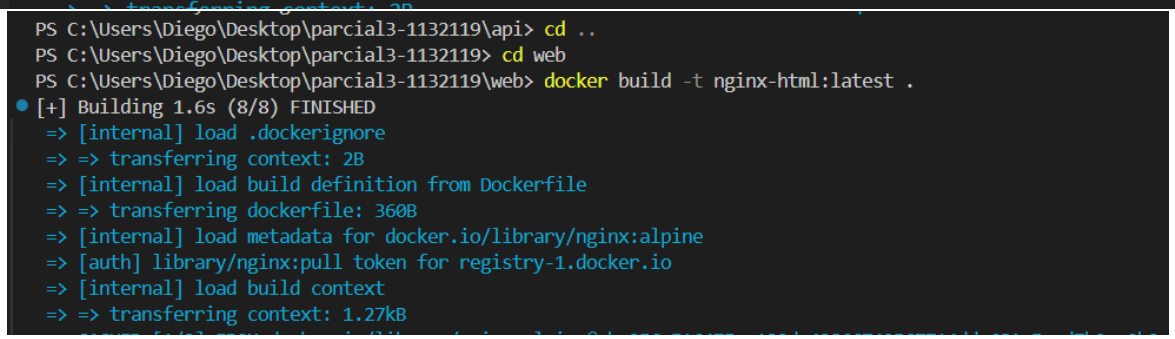
Parcial 3

Se utilizó Pyhton para la API y MongoDB para la base de datos no relacional
Se crearon las aplicaciones para su debido uso (Frontend y Backend)



Luego de eso, se creó su debido Dockerfile para cada servicio y se crearon las imágenes, para luego subirlo a Docker Hub





Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

```
Login Succeeded
PS C:\Users\Diego\Desktop\parcial3-1132119\web> docker tag nginx-html:latest ghandalf02/nginx-html:latest
PS C:\Users\Diego\Desktop\parcial3-1132119\web> docker push ghandalf02/nginx-html:latest
The push refers to repository [docker.io/ghandalf02/nginx-html]
e0a8413c707d: Pushed
ce495f7b0b7d: Layer already exists
9c70f446fbe2: Layer already exists
5be225e16e44: Layer already exists
3d04ead9b400: Layer already exists
af5598fef05f: Layer already exists
8fbd5a835e5e: Layer already exists
75061be64847: Layer already exists
d4fc045c9e3a: Layer already exists
latest: digest: sha256:1d315ea250e4684e434b4072fd8b24b89c8a651faaaeb3ebd1c2af8db5411e89 size: 2196
PS C:\Users\Diego\Desktop\parcial3-1132119\web>
```

```
PS C:\Users\Diego\Desktop\parcial3-1132119\api> docker tag python-api:latest ghandalf02/python-api:latest
PS C:\Users\Diego\Desktop\parcial3-1132119\api> docker push ghandalf02/python-api:latest
The push refers to repository [docker.io/ghandalf02/python-api]
93769ed52b9f: Layer already exists
b24a15efac13: Layer already exists
7d85747dd549: Layer already exists
4a2b9ad0ffeb: Layer already exists
ae96698df02c: Layer already exists
e555c0055a9b: Layer already exists
205262265e50: Layer already exists
146826fa3ca0: Layer already exists
5d4427064ecc: Layer already exists
latest: digest: sha256:48ca696d391e764aad0ee7b187070ac988df80a545f8e53ab2633057ab8f2a9f size: 2201
```

En este caso se usaron repositorio que ya se tenían creados recientemente para solo hacer el push al repositorio.

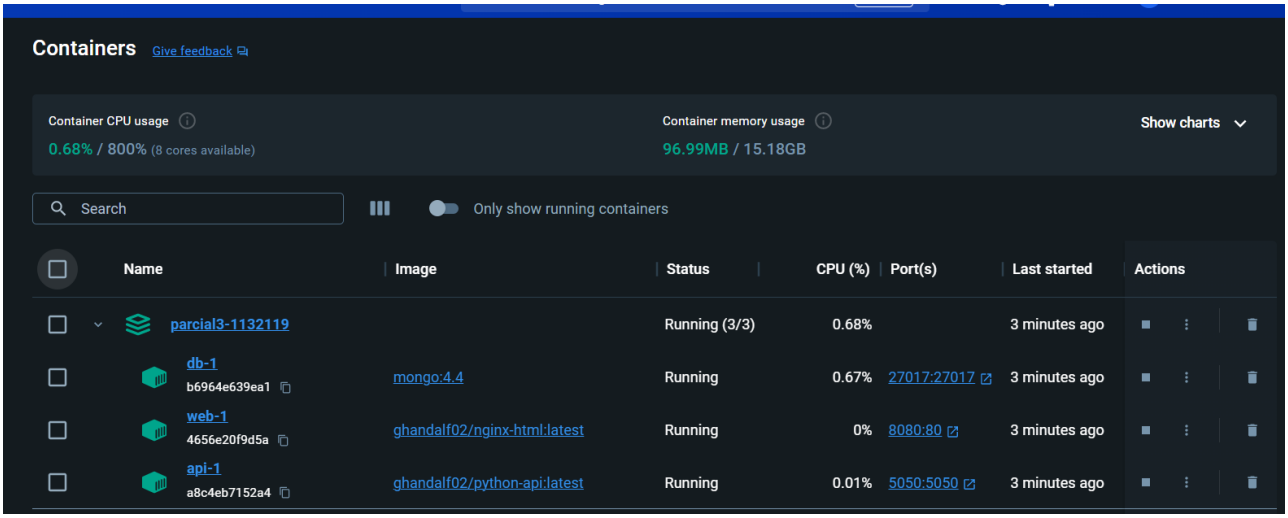
Después se creó el docker-compose para levantar los contenedores de Docker y verificar el funcionamiento.

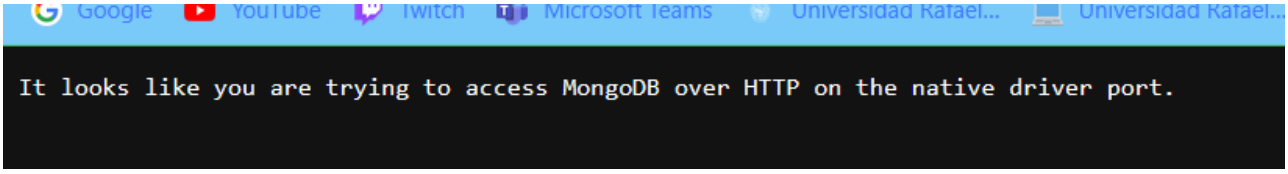
docker-compose.yml > {} services > {} web > [ ] volumes

```
docker-compose.yml - The Compose specification establishes a standard for the definition of multi-container platform-agnostic applications (compose-spec.json)
1  version: '3.8'
2  services:
3    db:
4      image: mongo:4.4
5      ports:
6        - "27017:27017"
7      volumes:
8        - mongo_data:/data/db
9    api:
10     image: ghandalf02/python-api:latest
11     ports:
12       - "5050:5050"
13     volumes:
14       - ./api:/app
15
16   web:
17     image: ghandalf02/nginx-html:latest
18     ports:
19       - "8080:80"
20     volumes:
21       - ./web:/usr/share/nginx/html
22 volumes:
23   mongo_data:
24
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

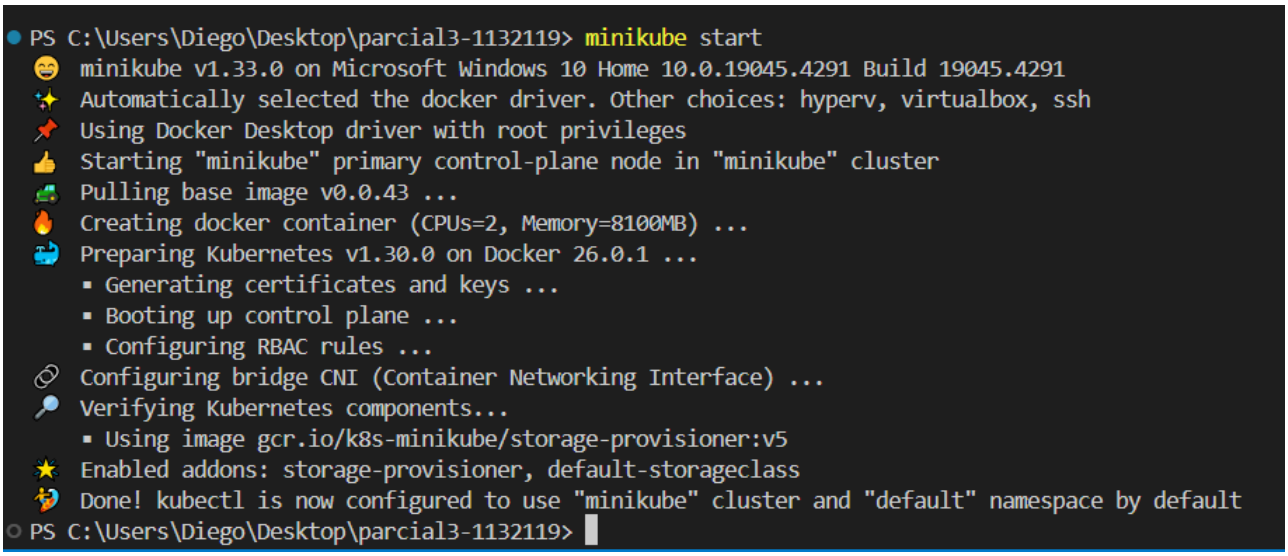
```
PS C:\Users\Diego\Desktop\parcial3-1132119\api> cd ..
PS C:\Users\Diego\Desktop\parcial3-1132119> docker-compose up -d
[+] Running 5/5
 ✓ Network parcial3-1132119_default       Created
 ✓ Volume "parcial3-1132119_mongo_data"   Created
 ✓ Container parcial3-1132119-api-1       Started
 ✓ Container parcial3-1132119-db-1        Started
 ✓ Container parcial3-1132119-web-1       Started
```





En esta imagen es cuando se quiere acceder a MongoDB cuando se creó el contenedor.

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119
Se utilizó Minikube para el espacio de kubernetes



Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

Luego de haber iniciado kubernetes, se procede a crear el main.tf para levantar la aplicación y exponerlo en un puerto. En el archivo de Terraform se crearon 3 servicios y 3 deployment, 1 para el backend, frontend y el servicio de base de datos. Para los primeros 2 se tomaron las imágenes que se subieron a Docker Hub respectivamente.

```
# Recurso para la implementación de frontend
resource "kubernetes_deployment" "nginx_html" {
  metadata {
    name = "nginx-html"
  }

  spec {
    replicas = 1
    selector {
      match_labels = {
        app = "nginx-html"
      }
    }

    template {
      metadata {
        labels = {
          app = "nginx-html"
        }
      }

      spec {
        container {
          image = "ghandalf02/nginx-html:latest"
          name  = "nginx-html"

          port {
            container_port = 80
          }
        }
      }
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

```
# Servicio para exponer aplicacion
resource "kubernetes_service" "nginx_html" {
  metadata {
    name = "nginx-html-service"
  }

  spec {
    selector = {
      app = "nginx-html"
    }

    port {
      port        = 8080
      target_port = 80
    }
  }
}
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

```
# Despliegue de MongoDB
resource "kubernetes_deployment" "mongodb" {
  metadata {
    name = "mongodb"
  }

  spec {
    replicas = 1
    selector {···
    }

    template {
      metadata {···
      }

      spec {
        container {
          image = "mongo:4.4"
          name  = "mongodb"

          port {
            container_port = 27017
          }

          volume_mount {
            mount_path = "/data/db"
            name       = "mongo-storage"
          }
        }

        volume {
          name = "mongo-storage"
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

```
# recurso para exponer el servicio de Mongo
resource "kubernetes_service" "mongodb_service" {
  metadata {
    name = "mongodb-service"
  }

  spec {
    selector = {
      app = "mongodb"
    }

    port {
      port        = 27017
      target_port = 27017
    }
  }
}
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

```
# Recurso para la implementación de la aplicación backend
resource "kubernetes_deployment" "pyhton_api" {
  metadata {
    name = "python-api"
    labels = {
      app = "python-api"
    }
  }

  spec {
    replicas = 1
    selector {
      match_labels = {
        app = "python-api"
      }
    }

    template {
      metadata {
        labels = {
          app = "python-api"
        }
      }

      spec {
        container {
          name  = "python-api"
          image = "ghandalf02/python-api:latest"
          port {
            container_port = 5050
          }
        }
      }
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119
Se inicializó Terraform y se aplicó el main.tf

```
PS C:\Users\Diego\Desktop\parcial3-1132119> terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/kubernetes from the dependency lock file
- Using previously-installed hashicorp/kubernetes v2.30.0

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.
PS C:\Users\Diego\Desktop\parcial3-1132119>
```

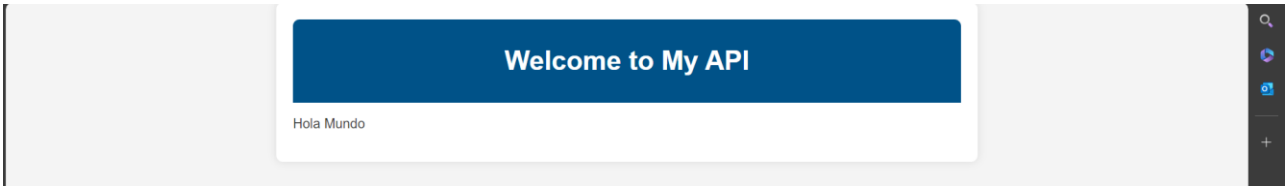
Se esperó a que se aplicaran los servicios y deployments.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

kubernetes_service.mongodb_service: Creating...
kubernetes_service.nginx_html: Creating...
kubernetes_service.pyhton_api: Creating...
kubernetes_deployment.nginx_html: Creating...
kubernetes_deployment.pyhton_api: Creating...
kubernetes_deployment.mongodb: Creating...
kubernetes_service.pyhton_api: Creation complete after 0s [id=default/python-api-service]
kubernetes_service.mongodb_service: Creation complete after 0s [id=default/mongodb-service]
kubernetes_service.nginx_html: Creation complete after 0s [id=default/nginx-html-service]
kubernetes_deployment.nginx_html: Creation complete after 8s [id=default/nginx-html]
kubernetes_deployment.pyhton_api: Still creating... [10s elapsed]
kubernetes_deployment.mongodb: Still creating... [10s elapsed]
kubernetes_deployment.pyhton_api: Creation complete after 16s [id=default/python-api]
kubernetes_deployment.mongodb: Still creating... [20s elapsed]
kubernetes_deployment.mongodb: Still creating... [30s elapsed]
kubernetes_deployment.mongodb: Still creating... [40s elapsed]
kubernetes_deployment.mongodb: Still creating... [50s elapsed]
kubernetes_deployment.mongodb: Creation complete after 56s [id=default/mongodb]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.
PS C:\Users\Diego\Desktop\parcial3-1132119>
```

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119

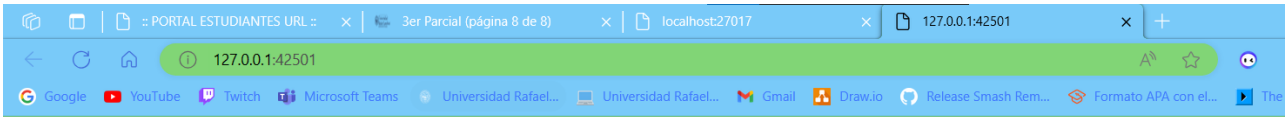
Luego de haber creado los servicios se aplica el siguiente comando: minikube service nginx-html-service –url; para conseguir el url y así poder acceder al servicio localmente. Se muestra de la siguiente forma ya con la aplicación Frontend

```
Apply complete! Resources: 6 added, 0 changed, 0 destroyed.
PS C:\Users\Diego\Desktop\parcial3-1132119> minikube service nginx-html-service --url
  service default/nginx-html-service has no node port
!  Services [default/nginx-html-service] have type "ClusterIP" not meant to be exposed, however for local development minikube allows you to access this !
http://127.0.0.1:42449
!  Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```



Vista de la aplicación Backend

```
PS C:\Users\Diego\Desktop\parcial3-1132119> minikube service python-api-service --url
  service default/python-api-service has no node port
!  Services [default/python-api-service] have type "ClusterIP" not meant to be exposed, however for local development minikube allows you to access this !
http://127.0.0.1:42501
!  Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```



Hola mundo

Universidad Rafael Landivar
Curso de Virtualización
Nombre: Diego Morales
Carnet: 1132119
Luego de esto se destruyeron los servicios, así mismo las imágenes para liberar espacio