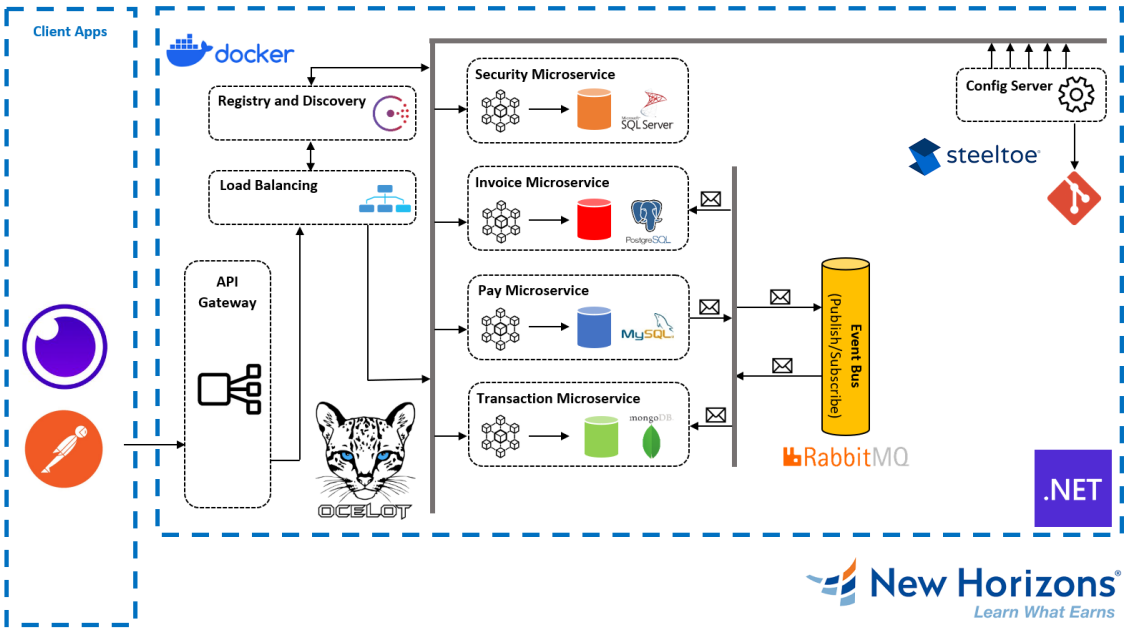
Muchik Market



Construir una solución con el nombre Muchik.Market y crear dentro de esta los siguientes proyectos:

- Muchik.Market.Security: Responsable del token de acceso
- Muchik.Market.Invoice: Responsable de listar las facturas de clientes
- Muchik.Market.Pay: Responsable de pagar una factura
- Muchik.Market.Transaction: Responsable de mostrar el detalle de una factura
- Muchik.Market.Gateway: Responsable de enmascarar las rutas del proyecto

NOTA: Compartir fuentes del proyecto, cada uno con su respectivo Docker File.

Crear las siguientes bases de datos:

- SQL SERVER: Para seguridad.
- POSTGRES: Para las facturas
- MYSQL: Para pagos
- MONGO: Para transacciones

Las bases de datos tendrán los siguientes modelos:

SQL SERVER

BD: db_security

CAMPOS:

id_user int primary key

username varchar 100

password varchar 100

POSTGRES

BD: db_invoice

CAMPOS: id_invoice

int primary key

amount decimal

state int

MYSQL

BD: db_operation

CAMPOS:

id_operation int primary key

id_invoice int amount decimal

date datetime

MONGO

BD: db_transaction

CAMPOS:

id_transaccion

id_invoice int

amount date

NOTA: Compartir los scripts de bases de datos

Crear las siguientes componentes:

- Una red de Docker llamada: muchik-nw
- Un servicio de configuración centralizada llamado: muchik-config
- Un servicio de colas llamado: muchik-bus
- Un servicio de registro y descubrimiento llamado: muchik-discovery
- Un servicio de balanceo llamado: muchik-balancer
- El microservicio de seguridad dockerizado se llamará: muchik -security
- El microservicio de invoces dockerizado se llamará: muchik -invoices
- El microservicio de pagos dockerizado se llamará: muchik -pay
- El microservicio de transacciones dockerizado se llamará: muchik-transaction
- El Gateway dockerizado se llamará: muchik -gateway
- Todos los servicios deben correr sobre la red muchik-nw.
- Compartir un Docker compose con todos los servicios
- Compartir un archivo con la configuracion de los enviroment del config server

Flujo del proyecto

- El microservicio de seguridad debe proporcionar un token de acceso para los microservicios de invoices, pay y transacciones.
- El gateway debe enmascarar las rutas de los servicios y debe sobrecargar la seguridad, además de enviar la solicitud al balanceador de carga para que este determine a que contenedor enviara la información, según su carga y disponibilidad. Todas las rutas deben trabajarse con el balanceador.
- El microservicio de invoices, debe listar las facturas de clientes y además debe consumir una cola para cambiar el estado de la factura cuando esta se paga a través del microservicio de pago.
- El microservicio de pago debe registrar el pago en su respectiva base de datos y además debe dejar un mensaje en una cola para actualizar la factura en el microservicio de facturas y además debe dejar un mensaje en una cola para registrar el movimiento en el microservicio de transacciones.
- El microservicio de transacciones debe listar las transacciones de una factura, además debe consumir una cola para obtener las transacciones de pago del microservicio de pago.
- Todos los microservicios deben consumir la cadena de conexión desde el servicio de configuración centralizada.