

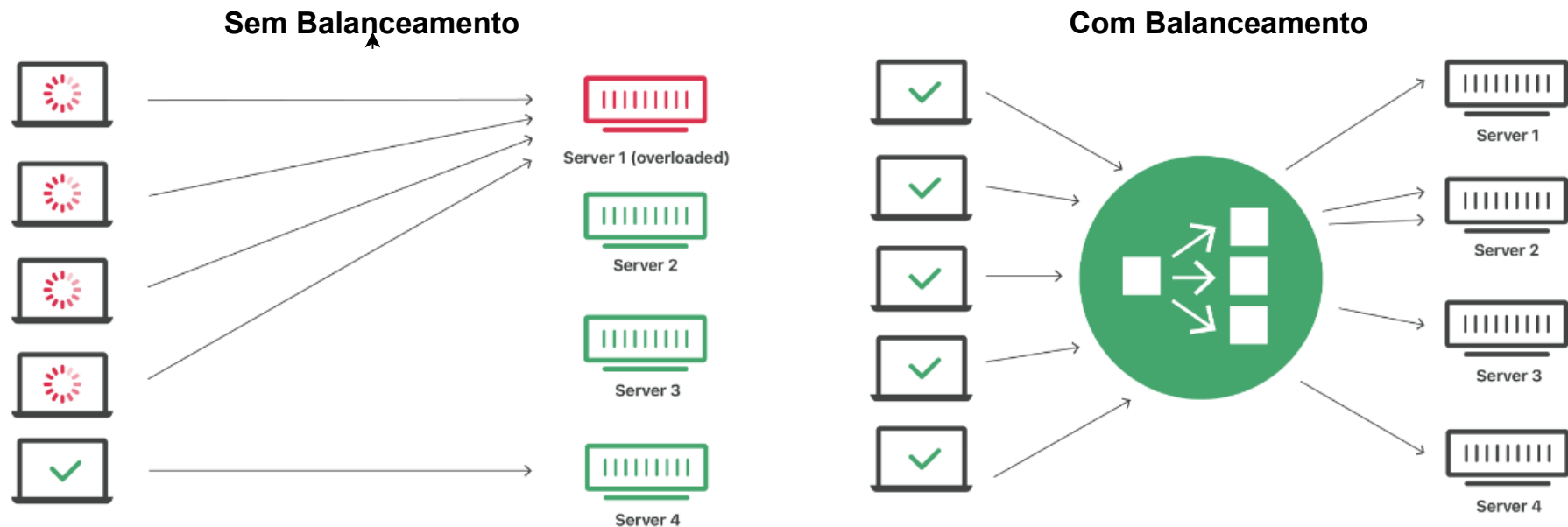
Plataforma Docker

Prof. Dr. Jesmmer Alves*

Instituto Federal Goiano Campus Morrinhos
Curso Bacharelado em Ciência da Computação

Balanceamento de Carga

Balanceamento de carga é a prática de distribuir cargas de trabalho computacionais entre dois ou mais computadores. Na internet, o balanceamento de carga é frequentemente empregado para dividir o tráfego de rede entre vários servidores. Isso reduz a sobrecarga em cada servidor e torna os servidores mais eficientes, acelerando a performance e reduzindo a latência. O balanceamento de carga é essencial para que a maioria dos aplicativos da internet funcione corretamente.

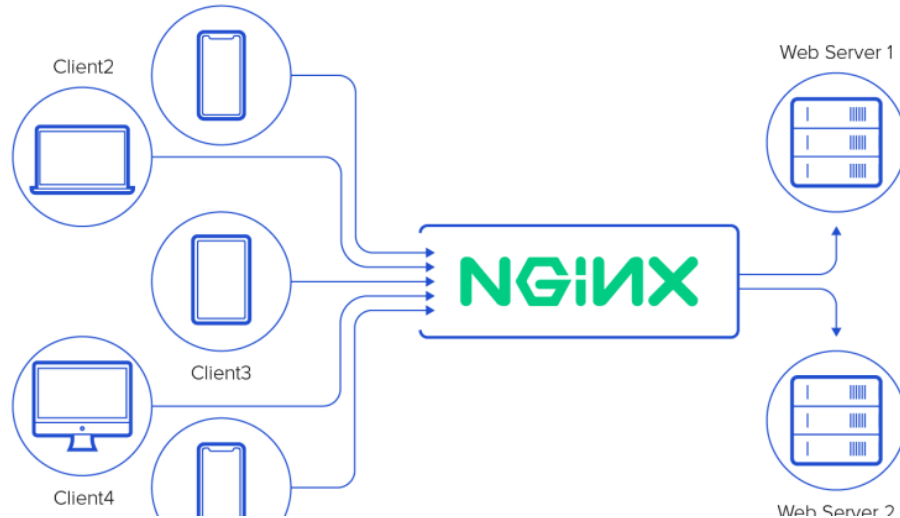


www.cloudflare.com

Aplicação 3 - Balanceamento de Carga com NGINX

O **NGINX** (Engine X) é um Software Open Source que tem como funcionalidade principal atender requisições HTTP na web.

Porém ele não serve somente para isso, não é atoa que ele é o webserver mais utilizado na internet. Além de atender requisições HTTP de forma excepcional existe uma serie de recursos que o torna cada vez mais interessante, sendo elas: proxy reverso, armazenamento em cache, balanceamento de carga, streaming de mídia e muitas vezes utilizado para funcionar como um servidor proxy para e-mail (IMAP, POP3 e SMTP). Mais a frente falo um pouco sobre cada uma dessas funcionalidades do nginx e em quais contextos são mais utilizadas.





Aplicação 3 - Balanceamento de Carga com NGINX


Vamos modificar a aplicação com Node js da aula anterior incluindo o balanceamento de carga com NGINX. Vamos precisar do seguinte sistema de arquivos. As aplicações são cópias idênticas da aplicação da aula anterior. A pasta nginx contém os arquivos para configuração do servidor NGINX. O arquivo docker-compose.yml cria a rede de containers para nossa aplicação.

▼ PROJETO NUM_VISITAS - AULA 3


▼ aplicacao-1


 Dockerfile


 package.json

 server.js


▼ aplicacao-2


 Dockerfile


 package.json

 server.js

▼ nginx

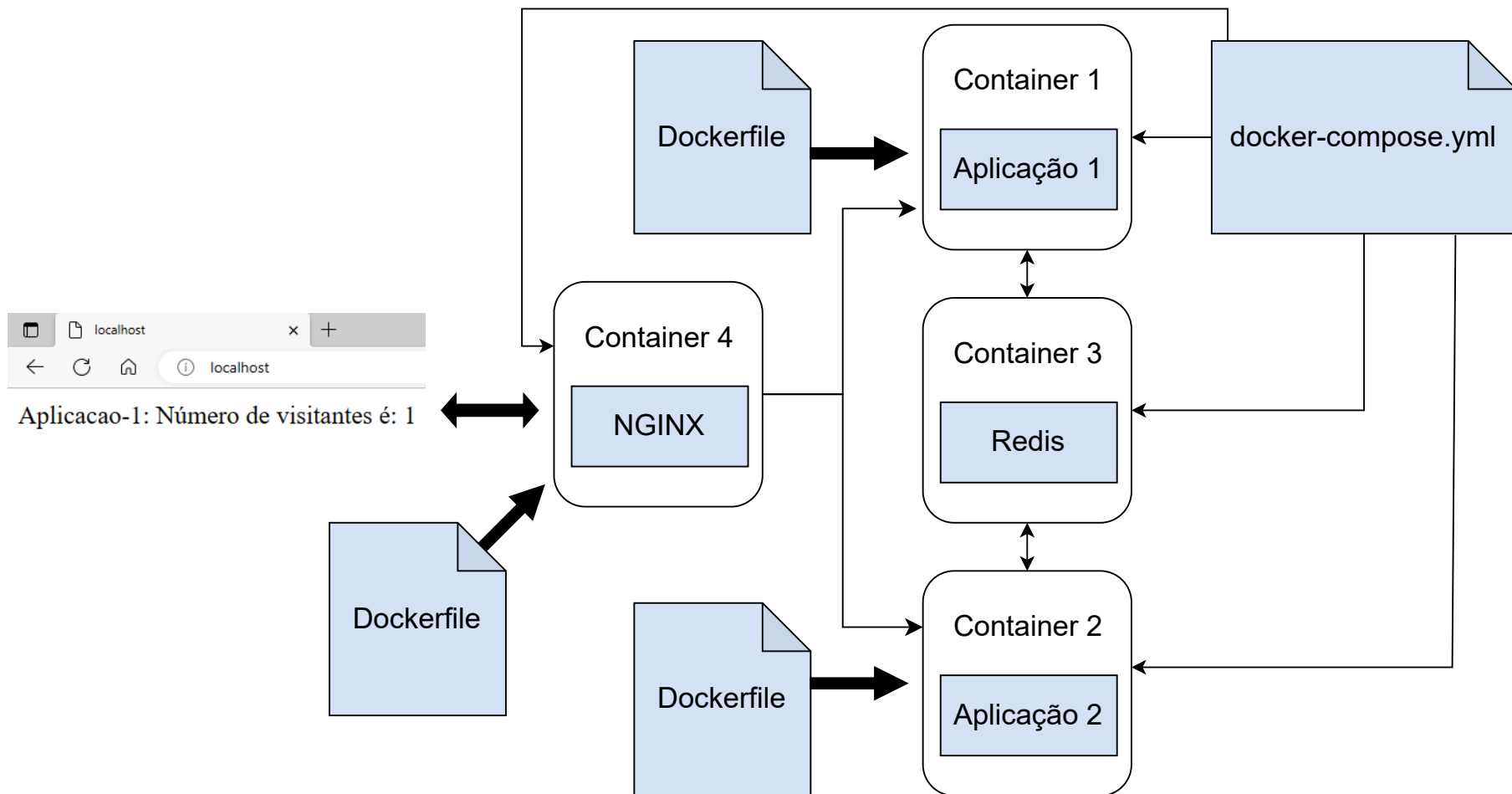
 Dockerfile

 nginx.conf

 docker-compose.yml

Aplicação 3 - Balanceamento de Carga com NGINX

Diagrama dos componentes



Aplicação 3 - Balanceamento de Carga com NGINX

aplicacao-1

Dockerfile

```
FROM node:alpine
WORKDIR /usr/src/app

COPY ./package*.json ./
RUN npm install
COPY ./server.js ./

CMD ["npm", "start"]
```

package.json

```
{
  "name": "aplicacao-1",
  "version": "1.0.0",
  "description": "Node.js + Express",
  "main": "server.js",
   Depurar
  "scripts": {
    "start": "node server.js"
  },
  "dependencies": {
    "express": "^4.17.2",
    "redis": "3.1.2"
  },
  "author": "",
  "license": ""
}
```

server.js

```
const express = require('express');
const redis = require('redis');
const app = express();
const redisClient = redis.createClient({
  host: 'test-redis',
  port: 6379
});

app.get('/', function(req, res) {
  redisClient.get('numVisits', function(err, numVisits) {
    numVisitsToDisplay = parseInt(numVisits) + 1;
    if (isNaN(numVisitsToDisplay)) {
      numVisitsToDisplay = 1;
    }
    res.send('Aplicacao-1: Número de visitantes é: '
      + numVisitsToDisplay);
    numVisits++;
    redisClient.set('numVisits', numVisits);
  });
});

app.listen(5000, function() {
  console.log('Web app está escutando porta 5000');
});
```

Aplicação 3 - Balanceamento de Carga com NGINX

aplicacao-2

Dockerfile

```
FROM node:alpine
WORKDIR /usr/src/app

COPY ./package*.json ./
RUN npm install
COPY ./server.js ./

CMD ["npm", "start"]
```

package.json

```
{
  "name": "aplicacao-2",
  "version": "1.0.0",
  "description": "Node.js + Express",
  "main": "server.js",
  ▶ Depurar
  "scripts": {
    "start": "node server.js"
  },
  "dependencies": {
    "express": "^4.17.2",
    "redis": "3.1.2"
  },
  "author": "",
  "license": ""
}
```

server.js

```
const express = require('express');
const redis = require('redis');
const app = express();

const redisClient = redis.createClient({
  host: 'test-redis',
  port: 6379
});

app.get('/', function(req, res) {
  redisClient.get('numVisits', function(err, numVisits) {
    numVisitsToDisplay = parseInt(numVisits) + 1;
    if (isNaN(numVisitsToDisplay)) {
      numVisitsToDisplay = 1;
    }
    res.send('Aplicacao-2: Número de visitantes é: '
      + numVisitsToDisplay);
    numVisits++;
    redisClient.set('numVisits', numVisits);
  });
});

app.listen(5000, function() {
  console.log('Aplicacao-2 está escutando porta 5000');
});
```

nginx

Dockerfile

```
FROM nginx  
RUN rm /etc/nginx/conf.d/default.conf  
COPY nginx.conf /etc/nginx/conf.d/default.conf
```

nginx.conf

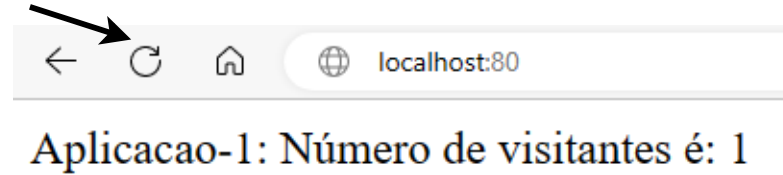
```
upstream balance-load {  
    server Aplicacao-1:5000;  
    server Aplicacao-2:5000;  
}  
  
server {  
    listen 80;  
    server_name localhost;  
    location / {  
        proxy_pass http://balance-load;  
    }  
}
```


Aplicação 3 - Balanceamento de Carga com NGINX

docker-compose.yml

```
version: '3.9'
services:
  test-redis:
    image: 'redis:alpine'
  aplicacao-1:
    restart: on-failure
    build: ./aplicacao-1
    ports:
      - '3008:5000'
  aplicacao-2:
    restart: on-failure
    build: ./aplicacao-2
    ports:
      - '3009:5000'
  nginx:
    build: ./nginx
    ports:
      - "80:80"
    depends_on:
      - aplicacao-1
      - aplicacao-2
```

```
docker-compose up --build
```



```
docker exec -it id_container sh
```

```
PS D:\Optativa BCC\Projeto num_visitas - aula 3> docker exec -it b4f1faeb9af9 sh
/data # redis-cli
127.0.0.1:6379> get numVisits
"1"
127.0.0.1:6379> |
```

1. Pesquise por um arquivo docker-compose.yml para usar o Airflow em containers docker.
 - a. Coloque o Airflow para funcionar no docker.
 - b. Faça uma descrição do arquivo docker-compose.yml usado e envie pelo Moodle.