

PRAKTIKUM A11
REVERSE PROXY AND LOAD BALANCING



DI SUSUN OLEH :

Nadya Indah Trisnawati (3122640034)
Mochammad Jauhar Ulul Albab (3122640044)
LJ D4 IT B

**PROGRAM STUDI TEKNIK INFORMATIKA DEPARTEMEN
TEKNIK INFORMATIKA DAN KOMPUTERPOLITEKNIK
ELEKTRONIKA NEGERI SURABAYA**

1. Reverse Proxy:

Reverse proxy adalah server yang bertindak sebagai perantara antara klien (pengguna) dan server tujuan yang sebenarnya. Ketika klien mengirim permintaan ke server, permintaan tersebut diteruskan melalui reverse proxy terlebih dahulu. Reverse proxy kemudian akan memproses permintaan tersebut dan meneruskannya ke server tujuan. Setelah server tujuan merespons, respons tersebut akan kembali melalui reverse proxy sebelum akhirnya diteruskan kepada klien.

Cara menggunakan reverse proxy dapat bervariasi tergantung pada lingkungan dan teknologi yang Anda gunakan. Berikut adalah langkah-langkah umum untuk mengimplementasikan konsep tersebut:

- a. Pilih dan instal server reverse proxy: Ada beberapa opsi server reverse proxy yang populer, seperti Nginx, Apache HTTP Server, atau HAProxy. Pilihlah yang sesuai dengan kebutuhan Anda dan instal di server yang ditentukan.
- b. Konfigurasi server reverse proxy: Setelah menginstal server reverse proxy, Anda perlu mengkonfigurasinya. Konfigurasi ini melibatkan penentuan server tujuan yang akan menerima permintaan yang diteruskan, mengatur aturan penyeimbangan beban jika diperlukan, serta pengaturan lainnya seperti caching dan keamanan.
- c. Uji coba dan validasi: Setelah konfigurasi selesai, lakukan uji coba dengan mengirim permintaan melalui reverse proxy dan memastikan bahwa respons dari server tujuan diterima dengan benar melalui reverse proxy.

2. Load Balancing:

Load balancing adalah proses mendistribusikan lalu lintas jaringan secara merata ke beberapa server tujuan untuk mencegah satu server menjadi terlalu terbebani. Tujuannya adalah untuk meningkatkan kinerja, ketahanan, dan skalabilitas sistem.

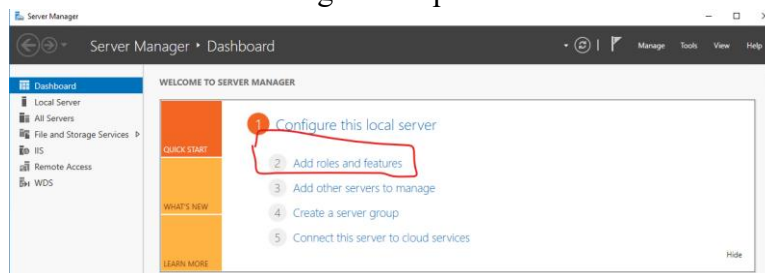
Terdapat beberapa metode load balancing yang umum digunakan:

- a. Round Robin: Permintaan diteruskan ke setiap server secara bergantian sesuai urutan.
- b. Least Connection: Permintaan diteruskan ke server dengan jumlah koneksi terendah pada saat itu.
- c. IP Hashing: Permintaan diteruskan berdasarkan alamat IP klien, sehingga permintaan dari klien yang sama selalu dikirim ke server yang sama.

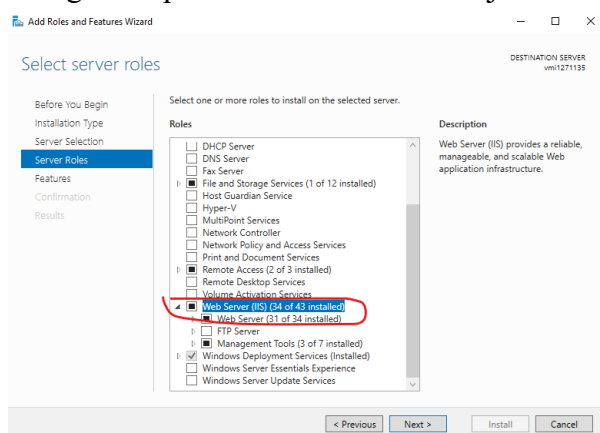
Load balancing dapat dilakukan dengan menggunakan perangkat keras khusus (seperti load balancer) atau menggunakan perangkat lunak (seperti reverse proxy). Penerapan load balancing membantu meningkatkan keandalan dan kinerja sistem dengan memastikan bahwa beban kerja terdistribusi secara seimbang di antara server-server yang tersedia.

Kedua konsep ini, reverse proxy dan load balancing, sering digunakan bersama-sama untuk meningkatkan kinerja, keandalan, dan keamanan infrastruktur server. Cara penerapan load balancing :

- a. Pilih pendekatan load balancing: Tentukan metode load balancing yang akan Anda gunakan, seperti round-robin, least connection, atau IP hashing. Setiap pendekatan memiliki kelebihan dan kelemahan tertentu, jadi pilihlah yang sesuai dengan kebutuhan dan lingkungan Anda.
 - b. Konfigurasi server load balancing: Jika Anda menggunakan perangkat keras load balancer, ikuti petunjuk pemasangan dan konfigurasi yang disediakan oleh produsen. Jika Anda menggunakan perangkat lunak atau server reverse proxy untuk melakukan load balancing, konfigurasikan aturan load balancing sesuai dengan pendekatan yang Anda pilih.
 - c. Tambahkan server tujuan: Tentukan server-server tujuan yang akan menerima lalu lintas yang didistribusikan dan tambahkan mereka ke dalam konfigurasi load balancer. Pastikan server-server tersebut berfungsi dengan benar dan siap menerima lalu lintas.
 - d. Uji coba dan validasi: Setelah konfigurasi selesai, uji coba dengan mengirim permintaan dan pastikan bahwa lalu lintas didistribusikan secara merata di antara server-server tujuan. Periksa juga apakah load balancing berfungsi seperti yang diharapkan dan jika ada kegagalan server, apakah permintaan masih bisa diarahkan ke server yang tersedia.
3. Langkah penerapan reverse proxy & load balancing pada windows server 2016
- Pastikan IIS telah diinstal:
 - Membuka "Server Manager" dan pilih "Add Roles and Features".

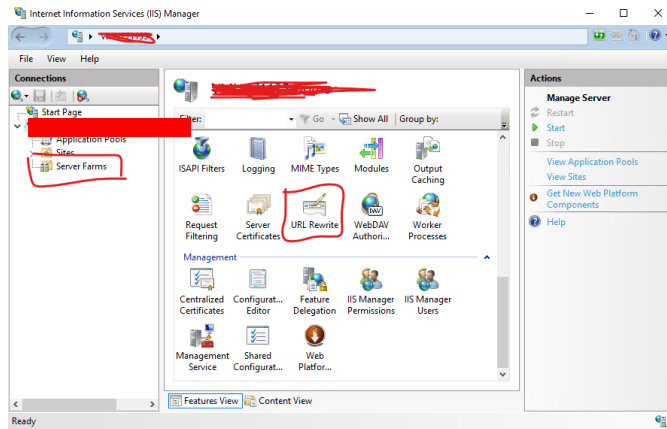


- Mengecek apakah IIS sudah terinstal jika belum instal terlebih dahulu

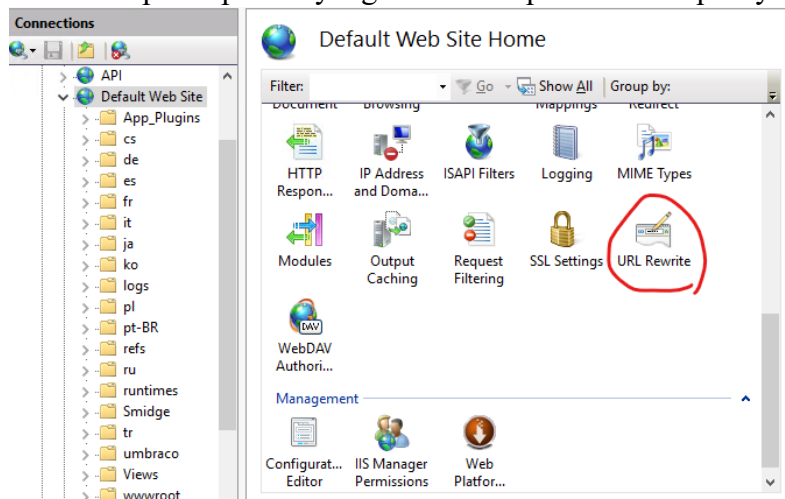


4. Install fitur yang dibutuhkan untuk melakukan reverse proxy
 - URL Rewrite (<https://www.iis.net/downloads/microsoft/url-rewrite>)
 - Application Request Routing (<https://www.iis.net/downloads/microsoft/application-request-routing>)

Setelah menginstall 2 modul tersebut maka akan muncul url rewrite pada moduli is, dan server farm

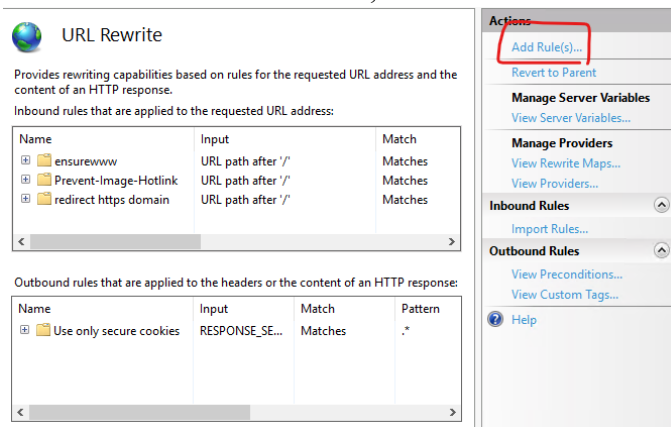


5. Atur Reverse Proxy pada pool aplikasi yang diinginkan
 - Masuk ke pool aplikasi yang akan diterapkan reverse proxy

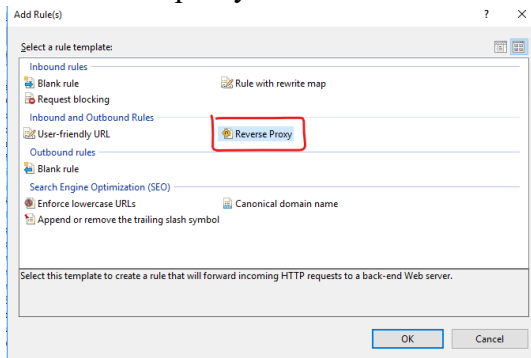


Setelah masuk ke menu aplikasi masuk ke menu url rewrite

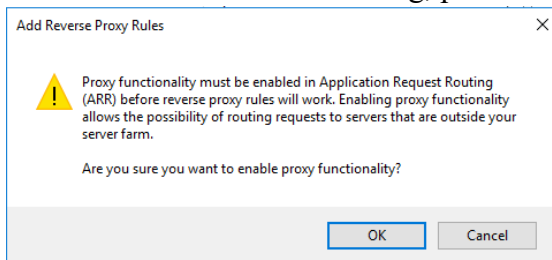
- Setelah masuk ke url rewrite, klik add rule



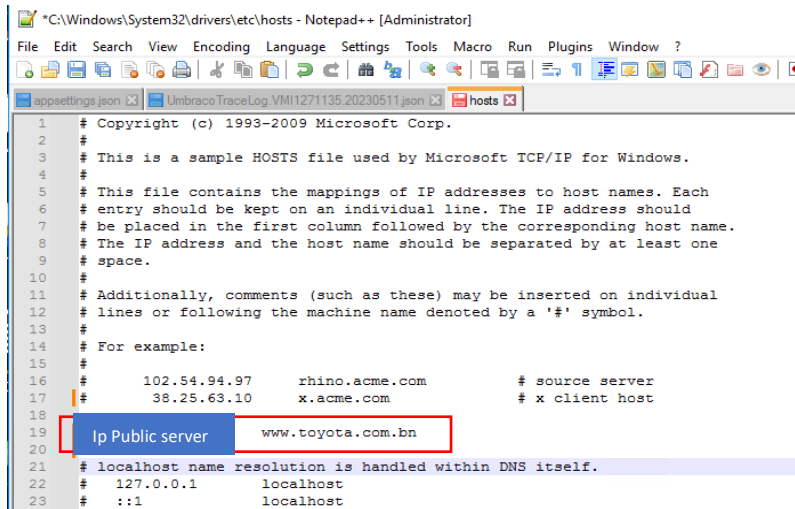
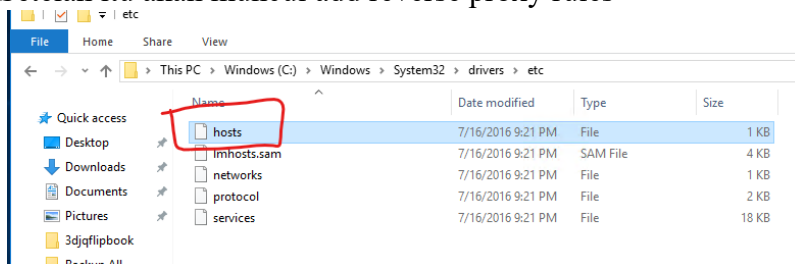
- Pilih reverse proxy



- Kemudian akan muncul warning, pilih ok



- Setelah itu akan muncul add reverse proxy rules



- Setelah itu isikan host yang telah ditambahkan pada record file hosts.

Add Reverse Proxy Rules

Inbound Rules
Enter the server name or the IP address where HTTP requests will be forwarded:
www.toyota.co.id
Example: contentserver1

☒ Enable SSL Offloading
Selecting this option will forward all HTTPS requests over HTTP.

Outbound Rules
☒ Rewrite the domain names of the links in HTTP responses
Responses that are generated by applications that are behind a reverse proxy can have HTTP links that use internal domain names. These links must be updated to use external domain names.
From:
Example: contentserver1
To:
Example: www.contoso.com

OK Cancel

- Rule reverse proxy berhasil ditambahkan

URL Rewrite

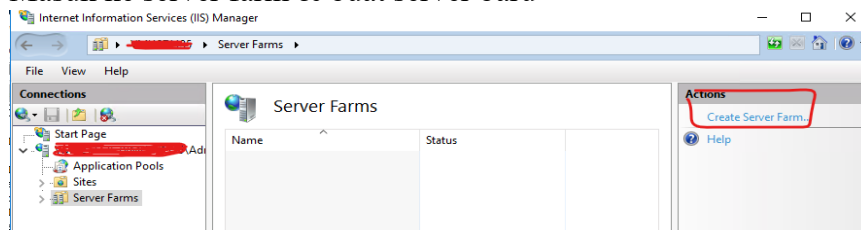
Provides rewriting capabilities based on rules for the requested URL address and the content of an HTTP response.

Inbound rules that are applied to the requested URL address:

Name	Input	Match
ensurewww	URL path after '/'	Matches
Prevent-Image-Hotlink	URL path after '/'	Matches
redirect https domain	URL path after '/'	Matches
ReverseProxyInboundRule1	URL path after '/'	Matches

6. Mengatur Load Balancing

- Masuk ke server farm & buat server baru



- Akan muncul wizard untuk menambahkan server farm, dan isikan nama server farm

Create Server Farm

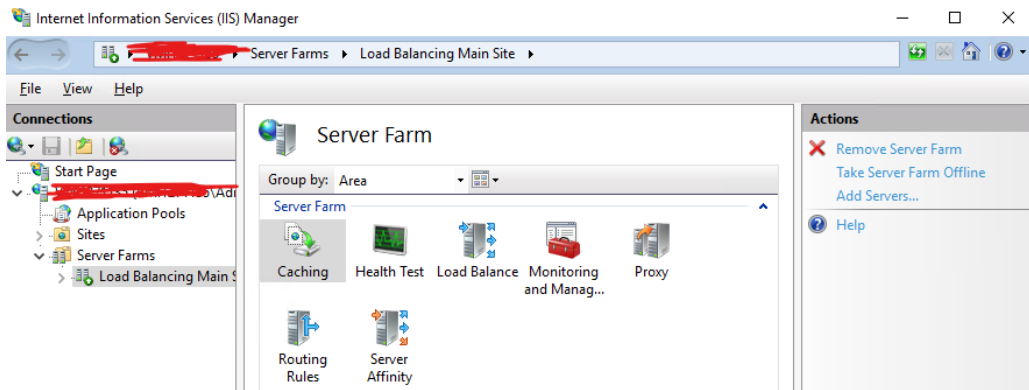
Specify Server Farm Name

Server farm name:
Load Balancing Main Site

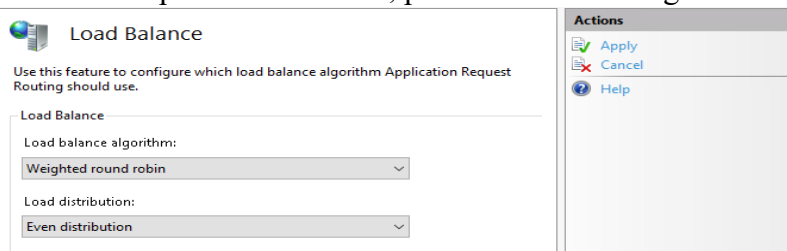
☒ Online

Previous Next Finish Cancel

- Setelah membuat server farm, input ip public dan selanjutnya akan diarahkan ke dashboard

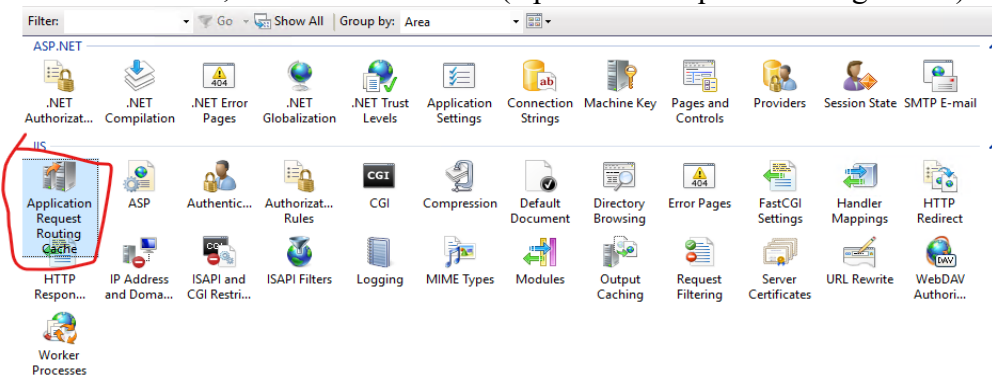


- Double klik pada load balance, pilih load balance algoritma

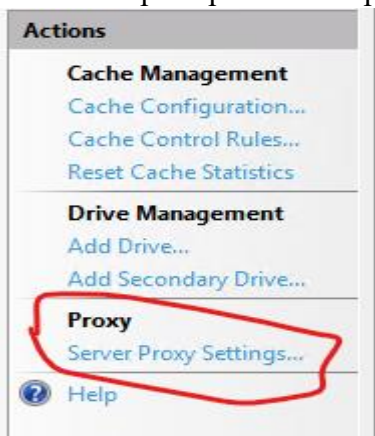


7. Enable IIS untuk berfungsi sebagai proxy

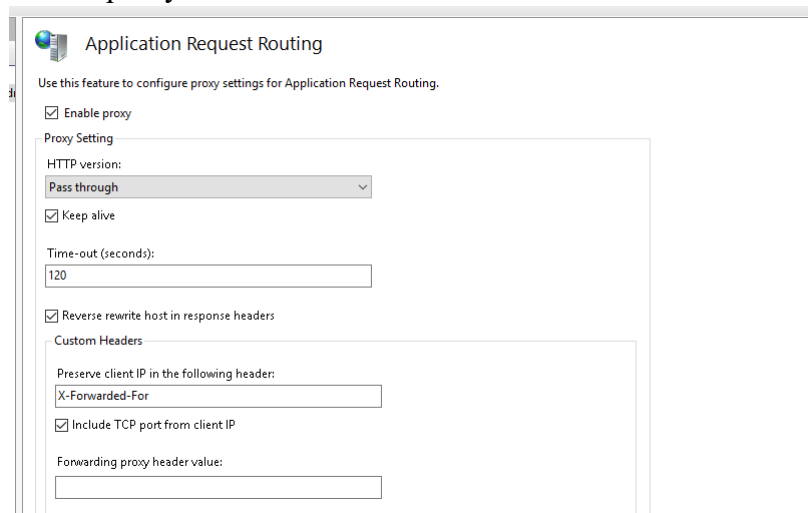
- Masuk ke root iis, dan cari menu ARR (Application Request Routing Cache)



- Kemudian pada pane action pilih server proxy setting



- Enable proxy



8. Reverse proxy pada linux
 - Membuat Aplikasi Node.js

```

root@kali: /var/www/html/nginx_server_project
File Actions Edit View Help

(root@kali)-[/var/www/html]
# mkdir nginx_server_project

(root@kali)-[/var/www/html]
# cd nginx_server_project

(root@kali)-[/var/www/html/nginx_server_project]
# pwd
/var/www/html/nginx_server_project

(root@kali)-[/var/www/html/nginx_server_project]
# npm init -y
Wrote to /var/www/html/nginx_server_project/package.json:

{
  "name": "nginx_server_project",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

```

9. membuat file server.js yang berisi source code untuk aplikasi.
Masukkan kode berikut ke dalam server.js. Di dalam server, akan dibuat dua respons berbeda, tergantung pada route saat ini. Kedua routes tersebut adalah /overview dan /api.

```

const http = require("http");

const server = http.createServer((req, res) => {
  const urlPath = req.url;
  if (urlPath === "/overview") {
    res.end("Welcome to the "overview page" of the nginx project");
  }
});

```

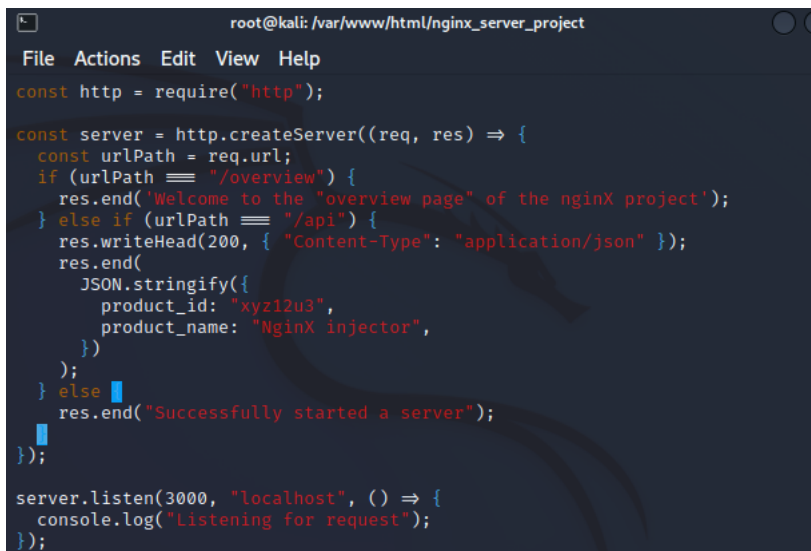


```

    } else if (urlPath === "/api") {
      res.writeHead(200, { "Content-Type": "application/json" });
      res.end(
        JSON.stringify({
          product_id: "xyz12u3",
          product_name: "NginX injector",
        })
      );
    } else {
      res.end("Successfully started a server");
    }
  });

  server.listen(3000, "localhost", () => {
    console.log("Listening for request");
  });

```



```

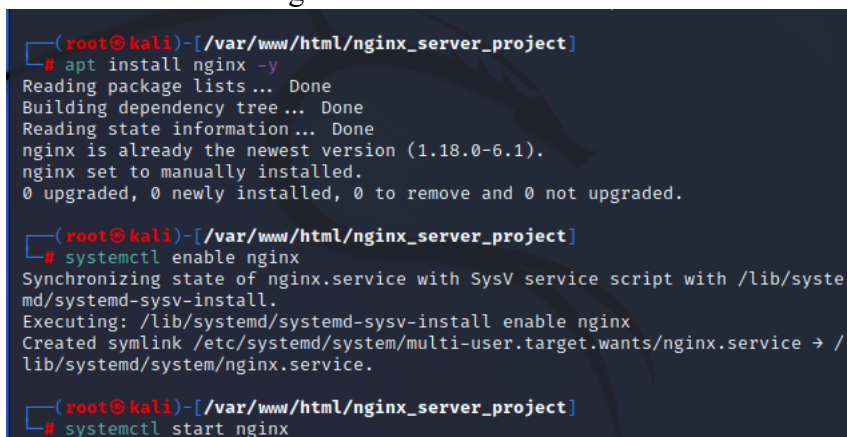
root@kali: /var/www/html/nginx_server_project
File Actions Edit View Help
const http = require("http");

const server = http.createServer((req, res) => {
  const urlPath = req.url;
  if (urlPath === "/overview") {
    res.end('Welcome to the "overview page" of the nginx project');
  } else if (urlPath === "/api") {
    res.writeHead(200, { "Content-Type": "application/json" });
    res.end(
      JSON.stringify({
        product_id: "xyz12u3",
        product_name: "NginX injector",
      })
    );
  } else {
    res.end("Successfully started a server");
  }
});

server.listen(3000, "localhost", () => {
  console.log("Listening for request");
});

```

10. Melakukan Instalasi nginx



```

(root@kali)-[/var/www/html/nginx_server_project]
# apt install nginx -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nginx is already the newest version (1.18.0-6.1).
nginx set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

(root@kali)-[/var/www/html/nginx_server_project]
# systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /lib/systemd/system/nginx.service.

(root@kali)-[/var/www/html/nginx_server_project]
# systemctl start nginx

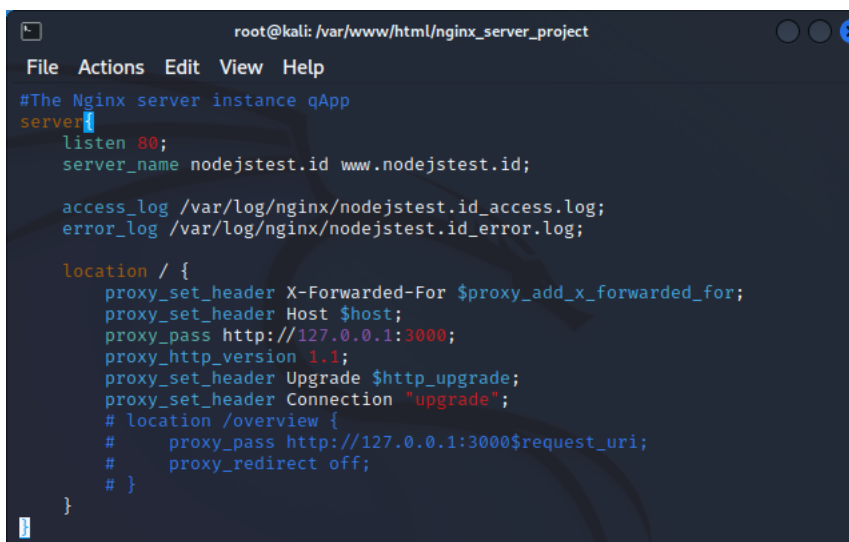
```

11. Melakukan konfigurasi Nginx reverse proxy

```
$ vim /etc/nginx/sites-available/nginx_server_project.conf
Masukkan kode berikut
#The Nginx server instance qApp
server{
    listen 80;
    server_name nodejstest.id www.nodejstest.id;

    access_log /var/log/nginx/nodejstest.id_access.log;
    error_log /var/log/nginx/nodejstest.id_error.log;

    location / {
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $host;
        proxy_pass http://127.0.0.1:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        # location /overview {
        #     proxy_pass http://127.0.0.1:3000$request_uri;
        #     proxy_redirect off;
        # }
    }
}
```



The screenshot shows a terminal window with the title bar "root@kali: /var/www/html/nginx_server_project". The window contains the same Nginx configuration code as the previous block, displayed in a dark-themed editor. The code is as follows:

```
File Actions Edit View Help
#The Nginx server instance qApp
server{
    listen 80;
    server_name nodejstest.id www.nodejstest.id;

    access_log /var/log/nginx/nodejstest.id_access.log;
    error_log /var/log/nginx/nodejstest.id_error.log;

    location / {
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $host;
        proxy_pass http://127.0.0.1:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        # location /overview {
        #     proxy_pass http://127.0.0.1:3000$request_uri;
        #     proxy_redirect off;
        # }
    }
}
```

12. Simpan dan silakan membuat symbolic untuk konfigurasi Nginx, lalu verifikasi konfigurasi Nginx dan silakan restart Nginx

```
(root@kali)-[/var/www/html/nginx_server_project]
# ln -s /etc/nginx/sites-available/nginx_server_project.conf /etc/nginx/sites-enabled/

(root@kali)-[/var/www/html/nginx_server_project]
# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

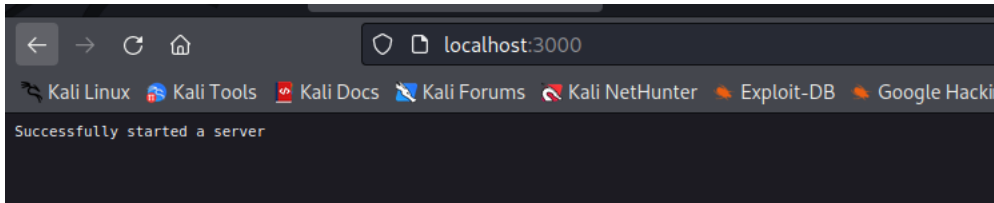
(root@kali)-[/var/www/html/nginx_server_project]
# systemctl restart nginx
```

13. Jalankan aplikasi Node.js menggunakan perintah node seperti berikut ini:

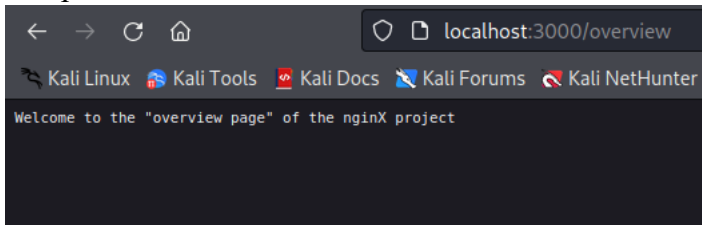
```
(root@kali)-[/var/www/html/nginx_server_project]
# cd /var/www/html/nginx_server_project/

(root@kali)-[/var/www/html/nginx_server_project]
# node server.js
```

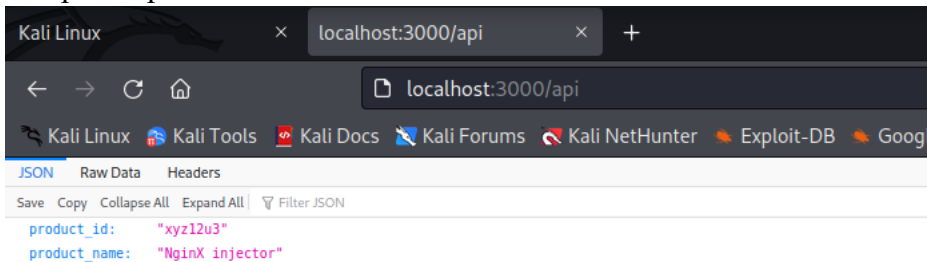
14. Ketikkan localhost:3000 di browser



Tampilan overview :



Tampilan api



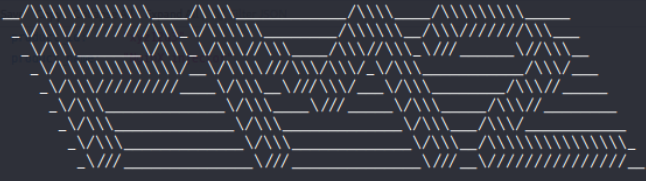
15. INSTALASI PM2

```
(root@kali)-[/var/www/html/nginx_server_project]
# npm install pm2 -g
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
/usr/local/bin/pm2 -> /usr/local/lib/node_modules/pm2/bin/pm2
/usr/local/bin/pm2-dev -> /usr/local/lib/node_modules/pm2/bin/pm2-dev
/usr/local/bin/pm2-runtime -> /usr/local/lib/node_modules/pm2/bin/pm2-runtime
/usr/local/bin/pm2-docker -> /usr/local/lib/node_modules/pm2/bin/pm2-docker
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@~2.3.2 (node_modules/pm2/node_modules/chokidar/node_modules/fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"linux","arch":"x64"})

+ pm2@5.3.0
added 184 packages from 182 contributors in 27.941s
```

16. Start PM2

```
(root@kali)-[/var/www/html/nginx_server_project]
# pm2 start server.js
```



Runtime Edition

PM2 is a Production Process Manager for Node.js applications with a built-in Load Balancer.

Start and Daemonize any application:
\$ pm2 start app.js

Load Balance 4 instances of api.js:
\$ pm2 start api.js -i 4

Monitor in production:
\$ pm2 monitor

Make pm2 auto-boot at server restart:
\$ pm2 startup

To go further checkout:
<http://pm2.io/>

[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /var/www/html/nginx_server_project/server.js in fork_mode (1 instance)
[PM2] Done.

id	name	mode	↻	status	cpu	memory
----	------	------	---	--------	-----	--------

Untuk melihat semua list aplikasi node.js yang sedang berjalan menggunakan perintah seperti pada gambar :

```
(root@kali)-[/var/www/html/nginx_server_project]
# pm2 list
```

id	name	namespace	version	mode	pid	uptime	↻	status	cpu	mem	user	watching
0	server	default	1.0.0	fork	4243	63s	0	online	0%	39.1mb	root	disabled