

Laporan Praktek Topik Khusus 1

Instalasi dan Konfigurasi Nginx, Golang, dan Pengujian API



SEMESTER VI

DISUSUN OLEH :

KURNIAWAN ALEXANDER

2211083030

DOSEN PENGAMPU :

YULHERNIWATI, S.Kom.,MT

YUNUS SUPRIADI WIJAYA

PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI PADANG

2025

A. Dasar Teori

1. Nginx

Nginx adalah server web sumber terbuka yang dikenal karena kinerja tinggi, stabilitas, dan efisiensinya dalam menangani lalu lintas yang tinggi. Selain sebagai server web, Nginx juga sering digunakan sebagai reverse proxy, load balancer, dan cache HTTP, menjadikannya pilihan utama untuk aplikasi dengan skala besar.

Arsitektur berbasis event-driven pada Nginx memungkinkan penanganan ribuan koneksi secara simultan dengan konsumsi sumber daya yang minimal. Karena kemampuannya dalam menangani permintaan dengan efisien, Nginx banyak digunakan untuk mendukung aplikasi web modern, termasuk layanan berbasis cloud dan sistem terdistribusi.

2. Golang (Go Language)

Go, yang dikembangkan oleh Google, adalah bahasa pemrograman sumber terbuka yang dirancang untuk efisiensi, kesederhanaan, dan keandalan. Dengan dukungan bawaan untuk konkurensi melalui goroutine dan channel, Go sangat ideal untuk pengembangan aplikasi web, sistem terdistribusi, dan aplikasi jaringan.

Fitur garbage collection otomatisnya mengurangi kompleksitas dalam pengelolaan memori, sementara sistem tipe statis membantu mendeteksi kesalahan lebih awal pada tahap kompilasi. Kinerja tinggi Go menjadikannya pilihan utama dalam membangun API dan layanan web, didukung oleh framework seperti Gin dan Echo yang menyederhanakan pengembangan aplikasi web.

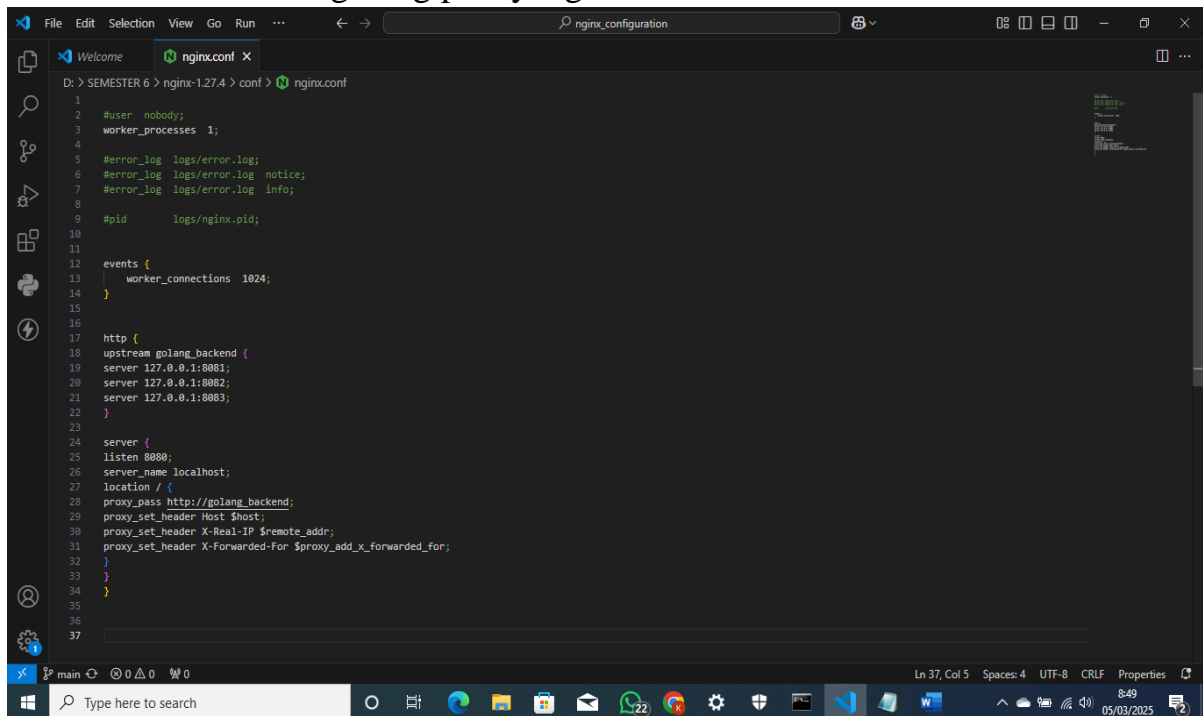
B. Tools

- Visual Studio Code
- Nginx
- Golang
- Command Prompt (CMD)

C. Langkah Kerja

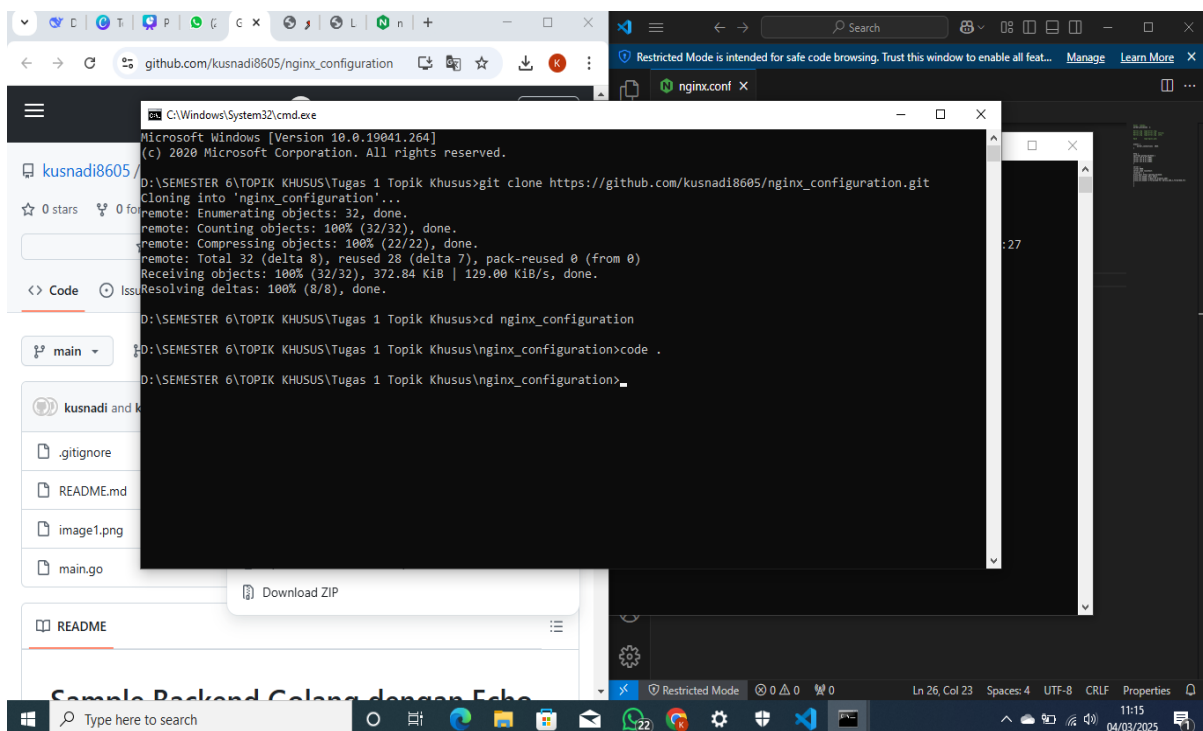
1. Instalasi dan Konfigurasi

- Download nginx
- Ekstrak dan jalankan nginx
- Edit file nginx.conf untuk melakukan konfigurasi menambah tiga instance golang port yang berbeda-beda.



```
1
2 #user  nobody;
3 worker_processes 1;
4
5 #error_log  logs/error.log;
6 #error_log  logs/error.log  notice;
7 #error_log  logs/error.log  info;
8
9 #pid        logs/nginx.pid;
10
11
12 events {
13     worker_connections 1024;
14 }
15
16
17 http {
18     upstream golang_backend {
19         server 127.0.0.1:8081;
20         server 127.0.0.1:8082;
21         server 127.0.0.1:8083;
22     }
23
24     server {
25         listen 8080;
26         server_name localhost;
27         location / {
28             proxy_pass http://golang_backend;
29             proxy_set_header Host $host;
30             proxy_set_header X-Real-IP $remote_addr;
31             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
32         }
33     }
34 }
35
36
37
```

- Lakukan git clone untuk menjalankan kode dari repository ke local



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19041.264]
(c) 2020 Microsoft Corporation. All rights reserved.

D:\SEMESTER 6\TOPIK KHUSUS\Tugas 1 Topik Khusus>git clone https://github.com/kusunadi8605/nginx_configuration.git
Cloning into 'nginx_configuration'...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 32 (delta 8), reused 28 (delta 7), pack-reused 0 (from 0)
Receiving objects: 100% (32/32), 372.84 KiB | 129.00 KiB/s, done.
Resolving deltas: 100% (8/8), done.

D:\SEMESTER 6\TOPIK KHUSUS\Tugas 1 Topik Khusus>cd nginx_configuration
D:\SEMESTER 6\TOPIK KHUSUS\Tugas 1 Topik Khusus\nginx_configuration>code .
D:\SEMESTER 6\TOPIK KHUSUS\Tugas 1 Topik Khusus\nginx_configuration>
```

-
- The screenshot shows a Windows terminal window with the following content:
- ```

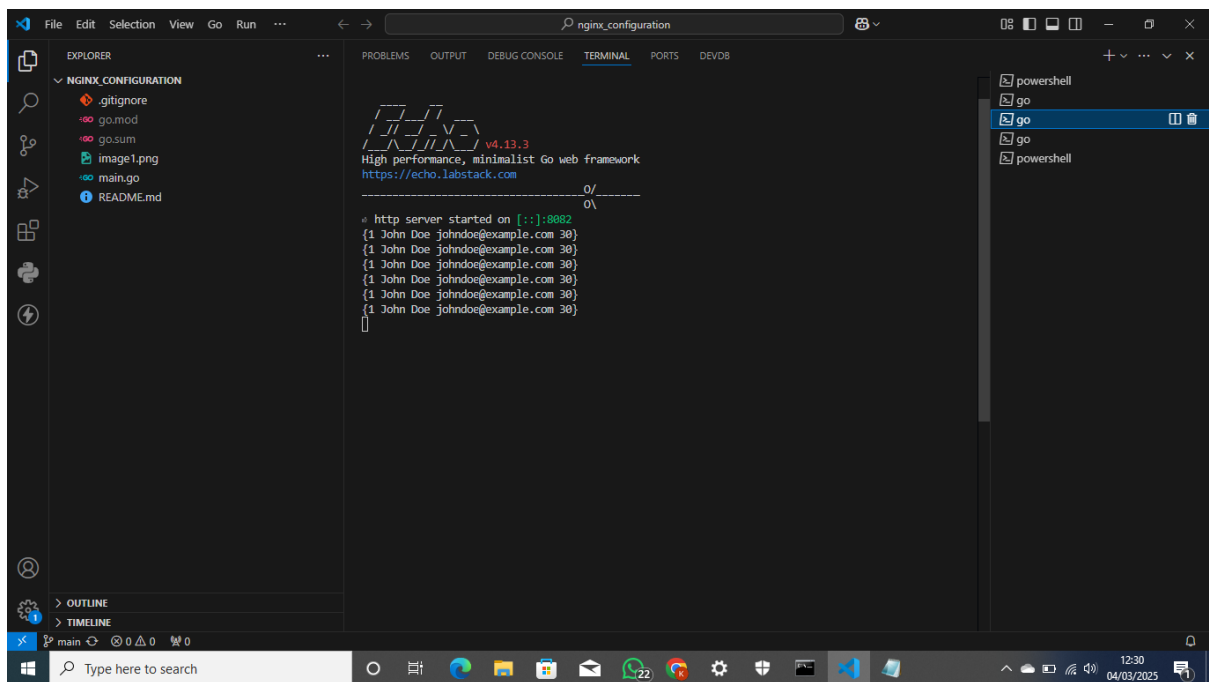
D:\> SEMESTER 6 > nginx-1.27.4 > conf > nginx.conf
1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS DEVD
History restored
go: to add module requirements and sums:
 go mod tidy
PS D:\SEMESTER 6\TOPIK KHUSUS\Tugas 1 Topik Khusus\ng
go: finding module for package github.com/labstack/ec
om/davech/go-spew v1.1.1
go: downloading github.com/valyala/fasttemplate v1
.2.2
go: downloading github.com/labstack/gommon v0.4.2
go: downloading github.com/pmezard/go-difflib v1.0
om/davech/go-spew v1.1.1
go: downloading github.com/valyala/fasttemplate v1.2.2
go: downloading github.com/labstack/gommon v0.4.2
go: downloading github.com/pmezard/go-difflib v1.0.0
go: downloading github.com/valyala/fasttemplate v1.2.2
go: downloading github.com/labstack/gommon v0.4.2
go: downloading github.com/pmezard/go-difflib v1.0.0
go: downloading golang.org/x/text v0.21.0
go: downloading golang.org/x/sys v0.28.0
go: downloading gopkg.in/yaml.v3 v3.0.1
go: downloading github.com/valyala/bytebufferpool v1.0.0
go: downloading github.com/pmezard/go-difflib v1.0.0
go: downloading golang.org/x/text v0.21.0
go: downloading golang.org/x/sys v0.28.0
go: downloading gopkg.in/yaml.v3 v3.0.1
go: downloading github.com/valyala/bytebufferpool v1.0.0
go: downloading golang.org/x/text v0.21.0
go: downloading golang.org/x/sys v0.28.0
go: downloading gopkg.in/yaml.v3 v3.0.1
go: downloading github.com/valyala/bytebufferpool v1.0.0
go: downloading github.com/valyala/bytebufferpool v1.0.0

```

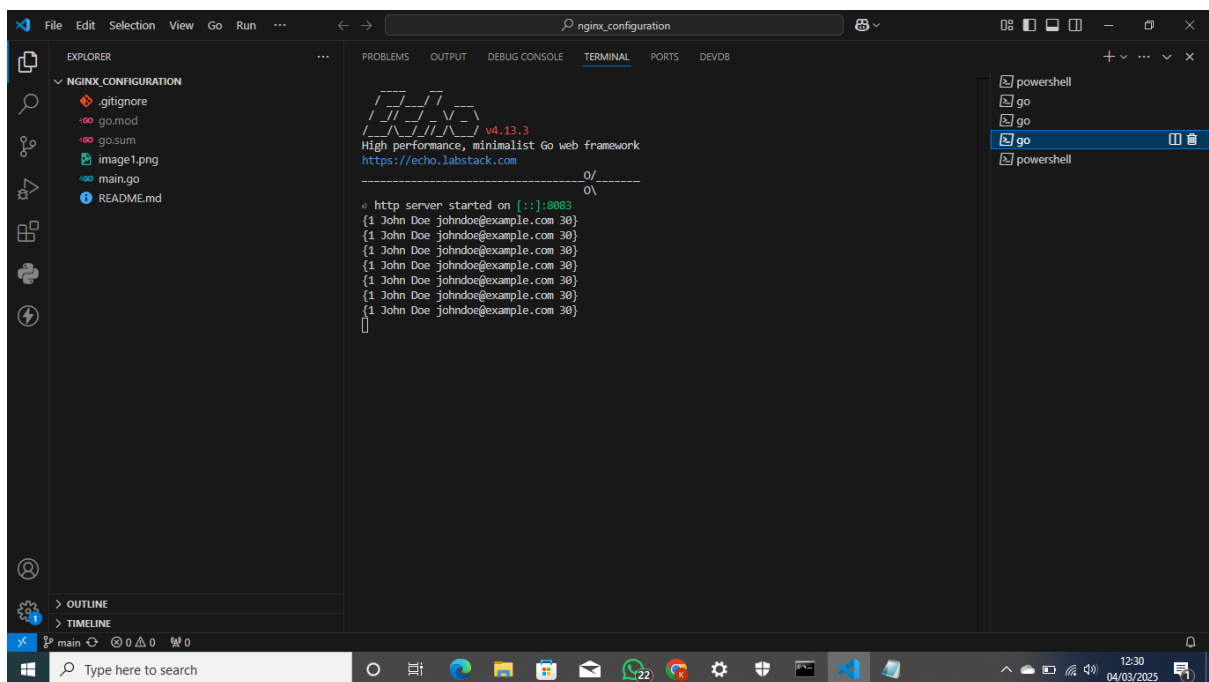
- Setelah melakukan instalasi dan konfigurasi sekarang file main.go sudah siap dijalankan pada tiga port sekaligus

- The image shows a screenshot of the Visual Studio Code editor interface. The Explorer pane on the left displays the project files: .gitignore, go.mod, go.sum, image1.png, main.go, and README.md. The main editor area shows the content of main.go, which includes a Go web framework implementation. The Terminal pane at the bottom shows the execution of 'go run main.go', which outputs 'High performance, minimalist Go web framework' and 'https://echo.labstack.com'. The Output pane on the right shows the command 'powershell' and its output 'go'.

- Port : 8082



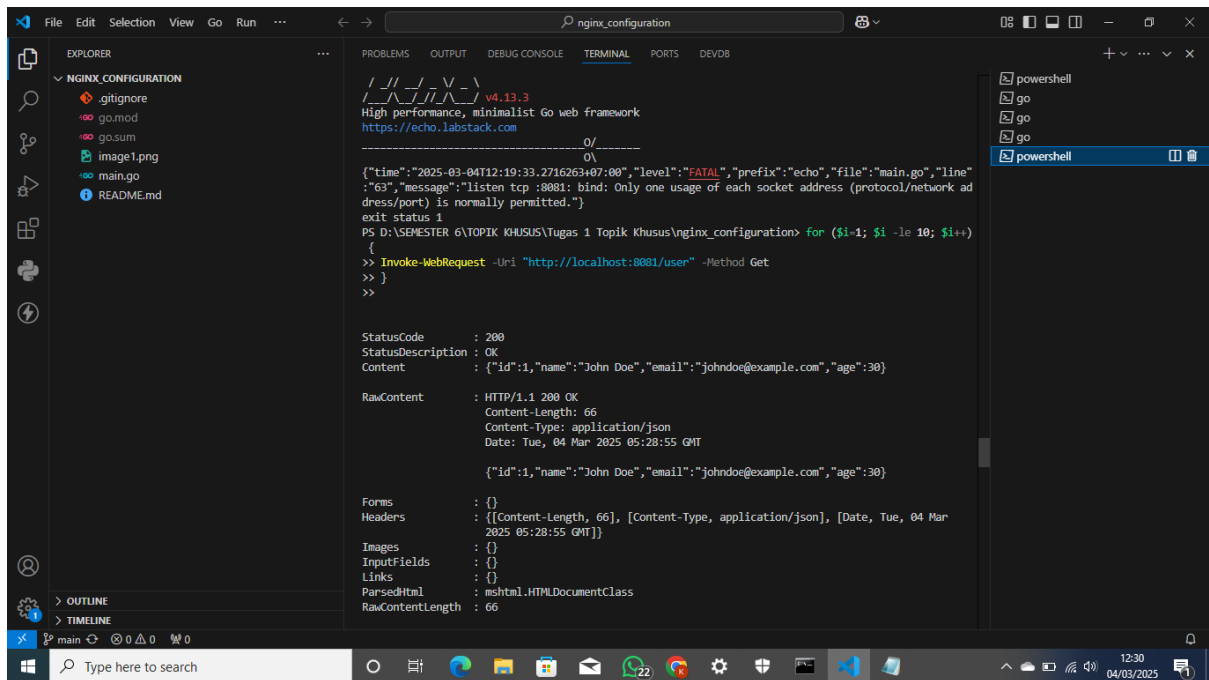
- Port : 8083



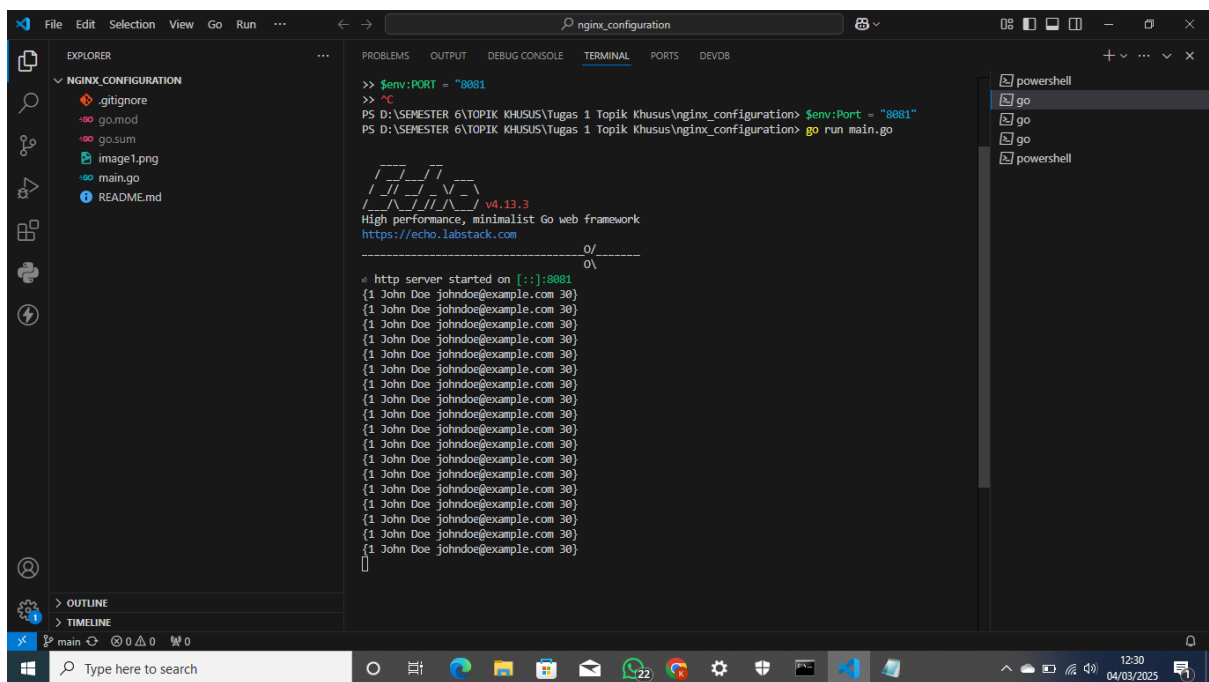
- Semua port berhasil dijalankan dengan sukses.

### 3. Pengujian API

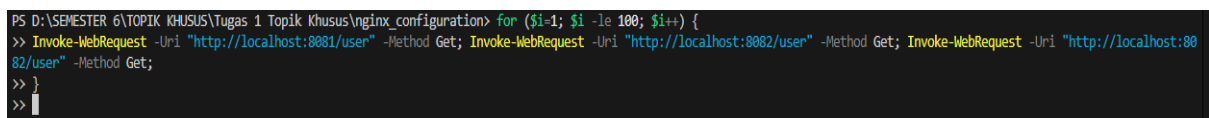
- Mengirim 10 permintaan Get ke API



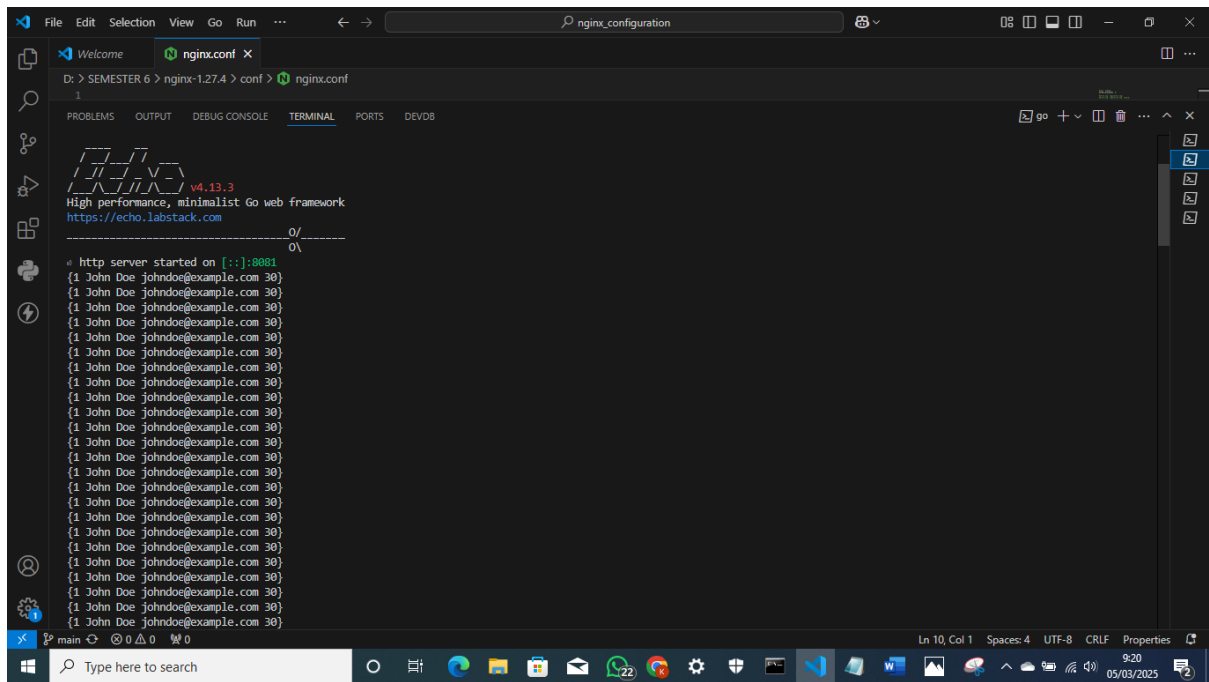
- Hasil dari get 10 kali pada port 8081



- Mengirim 100 permintaan Get API pada port

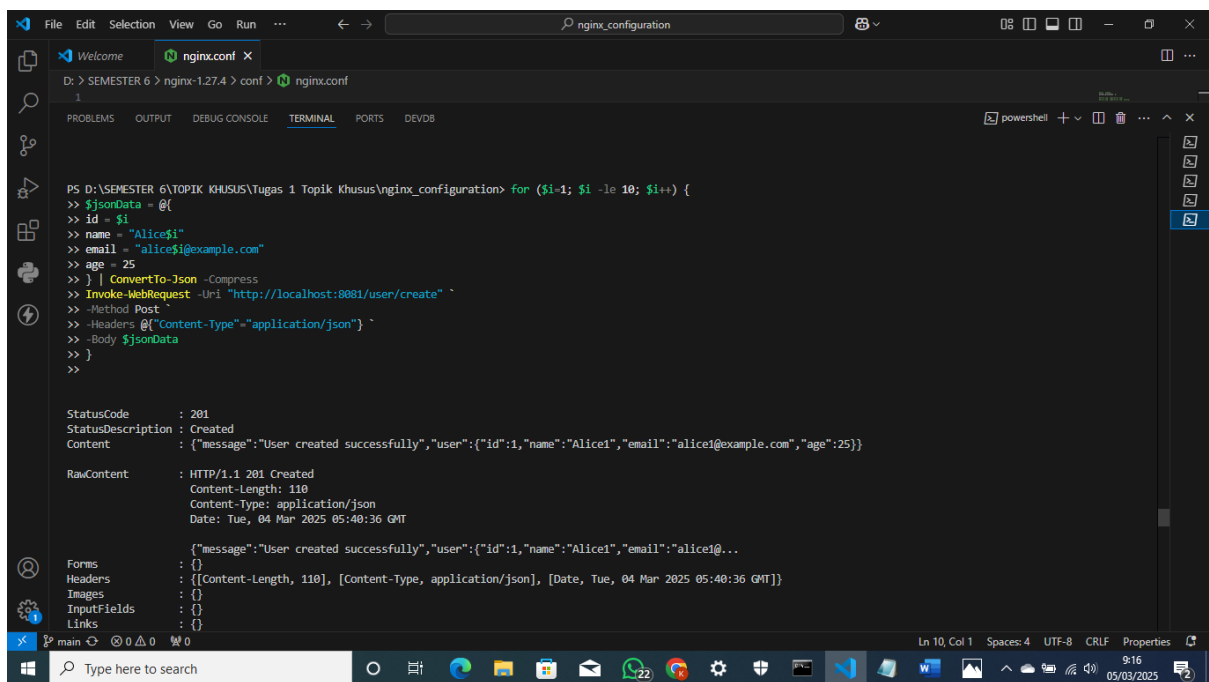


- Hasil dari permintaan 100 port



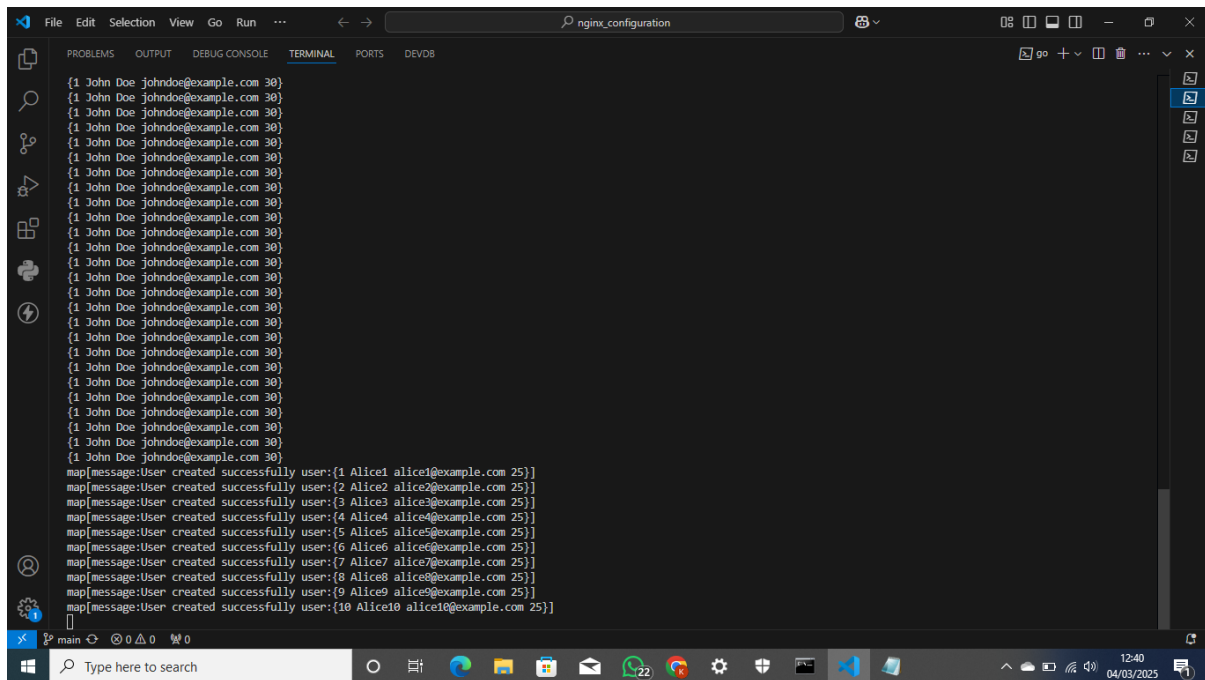
The screenshot shows a Visual Studio Code editor with a terminal window. The terminal output indicates that a Go web framework (v4.13.3) is running on port 8081. It shows a series of 10 POST requests from 'John Doe' to 'http://localhost:8081'. The requests are successful, and the response is '0/0'.

o Mengirim 10 permintaan Post ke API



The screenshot shows a Visual Studio Code editor with a PowerShell terminal window. The script sends 10 POST requests to 'http://localhost:8081/user/create' with a JSON body containing user information. The output of the first request is displayed, showing a 201 status code and a 'Created' status description. The response body is a JSON object: {"message": "User created successfully", "user": {"id": 1, "name": "Alice1", "email": "alice1@example.com", "age": 25}}.

o Hasil dari Post 10 kali pada port 8081



- Semua Langkah request Post dan Get API berhasil dijalankan tanpa error

## D. Kesimpulan

Berdasarkan praktik yang telah dilakukan, dapat disimpulkan bahwa integrasi antara Nginx dan Golang merupakan solusi yang optimal dalam membangun aplikasi web yang skalabel dan berkinerja tinggi. Konfigurasi Nginx sebagai load balancer, dengan beberapa instance aplikasi Golang yang berjalan pada port berbeda, berhasil menunjukkan kemampuannya dalam mendistribusikan beban secara efisien.

Selain itu, proses instalasi Golang di Linux serta pengujian API menggunakan PowerShell (GET & POST) membuktikan bahwa pengembangan dan pengujian aplikasi Golang dapat dilakukan dengan mudah. Secara keseluruhan, praktik ini memberikan wawasan yang lebih mendalam mengenai cara mengoptimalkan kombinasi Nginx dan Golang untuk membangun arsitektur aplikasi web yang andal dan efisien.