

**RANCANG BANGUN APLIKASI WEB KONVERSI SUHU  
MULTI SKALA BERBASIS GOLANG DENGAN  
ARSITEKTUR RESTFUL API**

Disusun guna memenuhi ujian akhir semester mata kuliah  
**KONTRUKSI PERANGKAT LUNAK**

**Dosen Pengampu :**

Yogi Kristiyanto, S. Kom, MMSi



**Disusun Oleh**

**Stevanus Andika Galih Setiawan(202303110008)**

**KELAS RK231**

**PROGRAM STUDI REKAYASA PERANGKAT**

**UNIVERSITAS IPWIJA**

**BOGOR**

**2025**

## KATA PENGANTAR

Puji syukur kehadiran Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, penulis dapat menyelesaikan makalah yang berjudul *"RANCANG BANGUN APLIKASI WEB KONVERSI SUHU MULTI SKALA BERBASIS GOLANG DENGAN ARSITEKTUR RESTFUL API"* tepat waktu. Makalah ini disusun sebagai salah satu syarat untuk memenuhi tugas akhir mata kuliah KONSTRUKSI PERANGKAT LUNAK di bawah bimbingan Bapak Yogi Kristiyanto, S.Kom., MMSi selaku dosen pengampu .

Ucapan terima kasih penulis sampaikan kepada:

1. Kedua orang tua yang senantiasa memberikan dukungan, doa, dan kasih sayang tanpa batas.
2. Bapak Yogi Kristiyanto, S.Kom., MMSi atas bimbingan, ilmu, dan motivasi yang diberikan selama proses pembelajaran.
3. Para idola/oshi, yaitu Raden Roro Freyanashifa Jayawardana dan Shania Gracia dari idol group JKT48 serta Nara dan Elma dari idol group Polaris yang menjadi sumber motivasi tak terhingga melalui karya dan dedikasinya, menginspirasi penulis untuk menyelesaikan makalah ini dengan semangat pantang menyerah.
4. Seluruh idol group, baik JKT48, Polaris, dan idol group lainnya yang tidak dapat disebutkan satu per satu, atas inspirasi dan energi positif yang terus mengalir melalui musik dan performa mereka.
5. Seluruh teman-teman, baik dari teman teman kelas RK231 maupun teman komunitas jejepangan yang tidak bisa disebutkan satu persatu, yang telah memberikan dukungan dan kebersamaan yang berarti.

Jakarta, 21 Juli 2025

Stevanus Andika Galih Setiawan

## DAFTAR ISI

### Contents

<b>KATA PENGANTAR.....</b>	<b>1</b>
<b>DAFTAR ISI.....</b>	<b>2</b>
<b>BAB I PENDAHULUAN.....</b>	<b>4</b>
1.1.    Indetifikasi Masalah .....	4
1.2.    Tujuan dan Manfaat Aplikasi.....	4
<b>BAB II LANDASAN TEORI .....</b>	<b>5</b>
2.1    Konsep Dasar Konversi Suhu .....	5
2.2    Rumus Konversi Suhu .....	5
Table 1.1 Rumus Konversi Suhu .....	5
2.3    Teknologi Pendukung .....	5
<b>BAB III PERANCANGAN SISTEM .....</b>	<b>6</b>
3.1.    Diagram Arsitektur.....	6
3.2.    DiagramAlir .....	8
3.3.    Desain Antarmuka.....	9
<b>BAB IV .....</b>	<b>10</b>
<b>IMPLEMENTASI DAN PENGUJIAN .....</b>	<b>10</b>
4.1 Struktur Modul/Program .....	10
.....	10
4.2 Penggunaan fungsi/metode .....	11
4.3    Penanganan Error .....	11
4.4    Dokumentasi Internal .....	13
4.5    Pengujian.....	15
<b>BAB V SARAN DAN KESIMPULAN .....</b>	<b>20</b>
5.1    Saran.....	20

5.2	Kesimpulan .....	21
<b>DAFTAR PUSTAKA.....</b>		<b>22</b>
<b>LAMPIRAN.....</b>		<b>23</b>

# **BAB I**

## **PENDAHULUAN**

### **1.1. Indetifikasi Masalah**

Dalam kehidupan sehari-hari, konversi suhu merupakan kebutuhan yang sering muncul, terutama bagi:

1. Pelajar dan akademisi yang mempelajari ilmu fisika.
2. Profesional di bidang meteorologi, kimia, dan Teknik.
3. Profesional di bidang meteorologi, kimia, dan Teknik.
4. Pengguna umum yang ingin memahami perbedaan skala suhu.

Adapun, masalah utama yang sering dihadapi adalah:

1. Kesulitan mengingat rumus konversi antar skala suhu.
2. Kesulitan mengingat rumus konversi antar skala suhu.
3. Kesulitan mengingat rumus konversi antar skala suhu.

### **1.2. Tujuan dan Manfaat Aplikasi**

Tujuan pembangunan aplikasi ThermoConvert:

1. Menyediakan alat konversi suhu yang akurat dan mudah digunakan.
2. Mengotomatisasi proses konversi antar skala suhu (Celsius, Fahrenheit, Kelvin).
3. Menyediakan antarmuka web dan API untuk fleksibilitas penggunaan.

Manfaat yang diharapkan:

1. Menghemat waktu dalam melakukan konversi suhu.
2. Meminimalisir kesalahan perhitungan manual.
3. Meningkatkan pemahaman tentang hubungan antar skala suhu.
4. Menyediakan solusi yang dapat diakses dari berbagai platform

## BAB II

### LANDASAN TEORI

#### 2.1 Konsep Dasar Konversi Suhu

Suhu dapat diukur dalam tiga skala utama:

Celsius (°C): Skala yang umum digunakan di sebagian besar negara, dengan titik beku air pada 0°C dan titik didih pada 100°C

Fahrenheit (°F): Umum digunakan di Amerika Serikat, dengan titik beku air pada 32°F dan titik didih pada 212°F

Kelvin (K): Skala absolut yang digunakan dalam ilmu pengetahuan, dengan 0 K sebagai nol absolut (-273.15°C)

#### 2.2 Rumus Konversi Suhu

Celsius ke Fahrenheit	Fahrenheit ke Celsius	Celsius ke Kelvin	Kelvin ke Celsius	Fahrenheit ke Kelvin	Kelvin ke Fahrenheit
$^{\circ}\text{F} = (^{\circ}\text{C} \times \frac{9}{5}) + 32$	$^{\circ}\text{C} = (^{\circ}\text{F} - 32) \times \frac{5}{9}$	$\text{K} = ^{\circ}\text{C} + 273.15$	$^{\circ}\text{C} = \text{K} - 273.15$	$\text{K} = (^{\circ}\text{F} - 32) \times \frac{5}{9} + 273.15$	$^{\circ}\text{F} = (\text{K} - 273.15) \times \frac{9}{5} + 32$

**Table 1.1 Rumus Konversi Suhu**

#### 2.3 Teknologi Pendukung

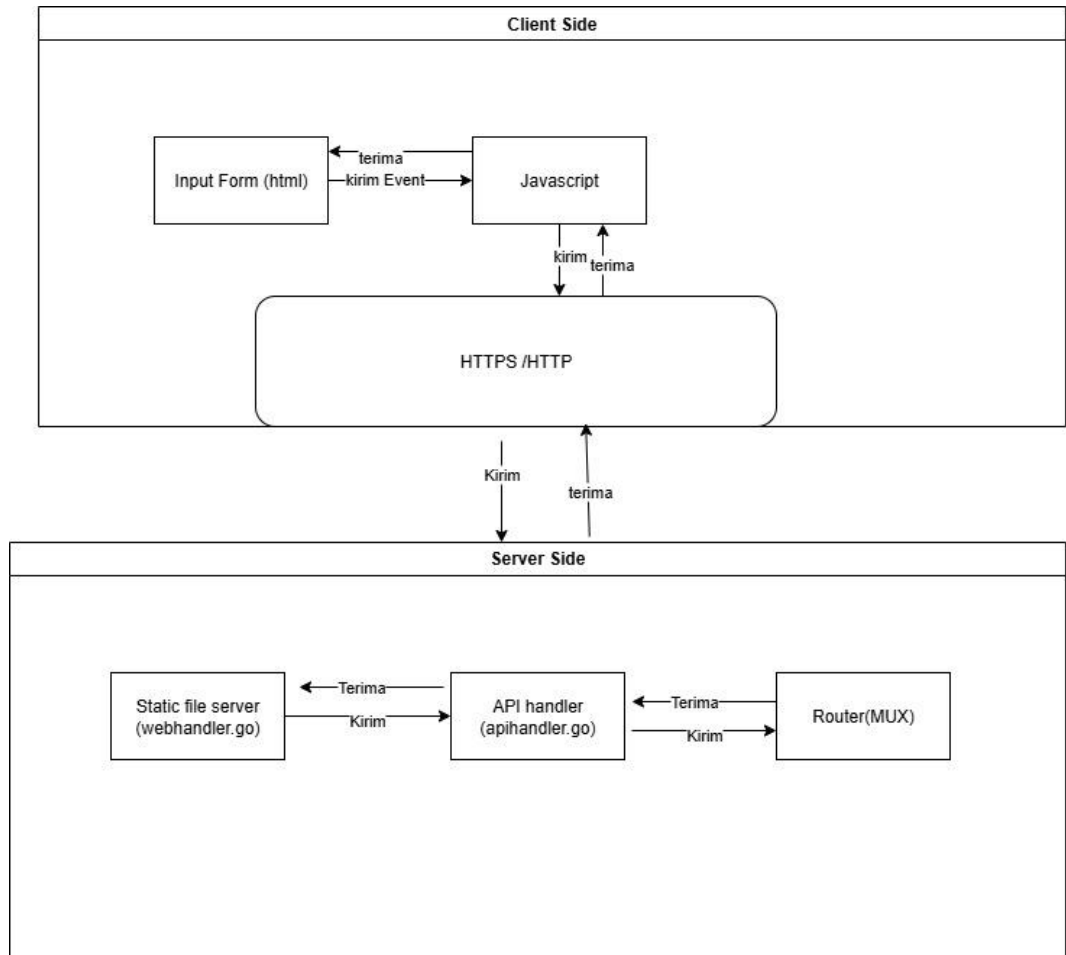
1. Golang: Bahasa pemrograman yang efisien untuk pengembangan aplikasi web dan API (Donovan & Kernighan, 2016)
2. Gorilla Mux: Router HTTP yang powerful untuk membangun aplikasi web dengan routing yang fleksibel (Gorilla Web Toolkit, 2023)
3. HTML/CSS/JavaScript: Teknologi standar untuk membangun antarmuka web interaktif.
4. Tailwind CSS: Framework CSS utility-first untuk pengembangan antarmuka yang responsif (Tailwind Labs, 2023)

## BAB III

### PERANCANGAN SISTEM

#### 3.1. Diagram Arsitektur

Adapun untuk diagram arsitektur adalah sebagai berikut :



Logika Diagram :

##### 1. Client Side:

- Dimulai dari Input Form (HTML) yang menerima input pengguna
- Terjadi Event (seperti submit form) yang memicu eksekusi JavaScript
- JavaScript melakukan fetch atau HTTP request ke server

## 2. Komunikasi:

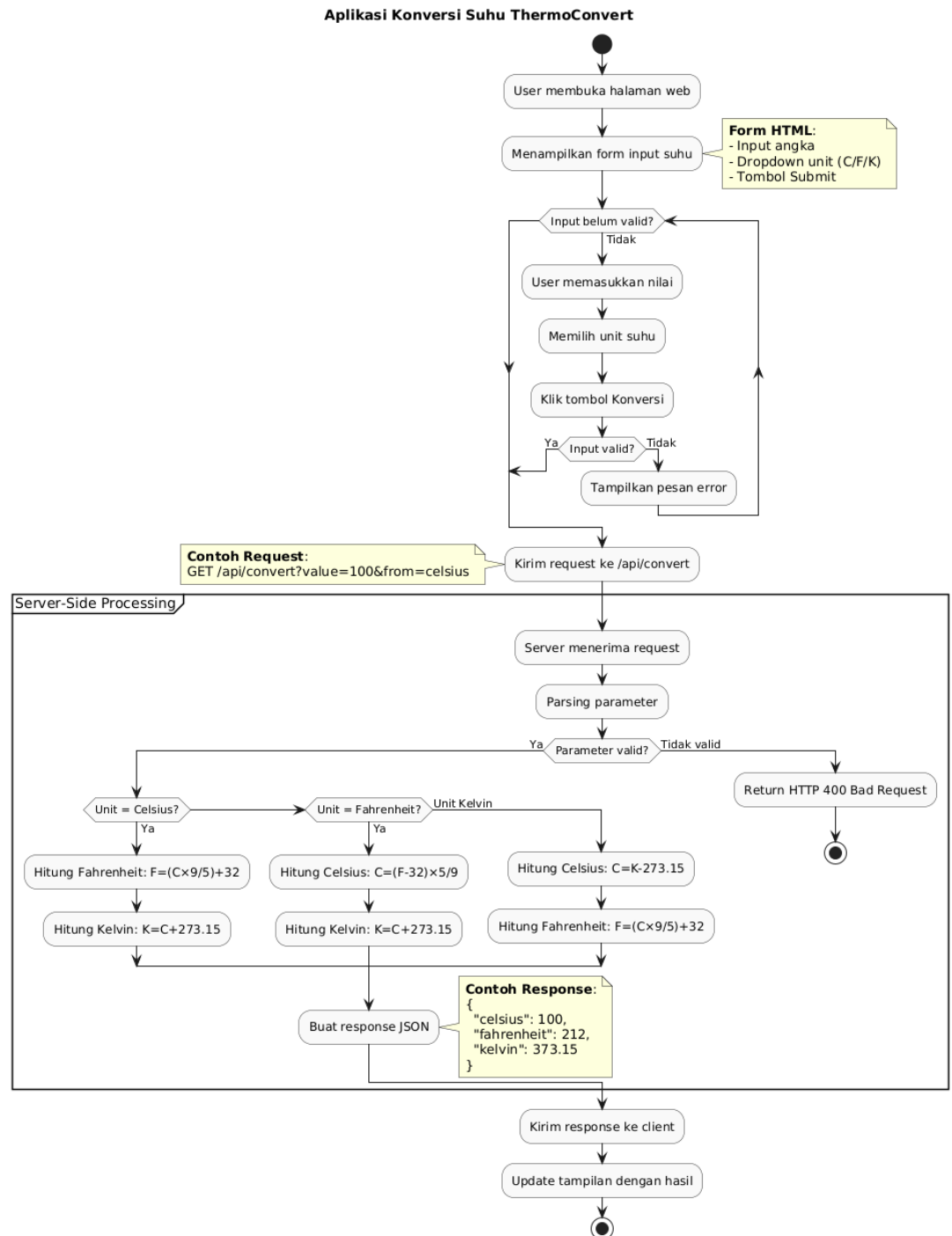
- Terdapat layer HTTPS/HTTP sebagai protokol komunikasi.
- Request dari client ke server

## 3. Server Side:

- Pertama diterima oleh Router (MUX) .
- Router mengarahkan request ke:
  - Static file server (webhandler.go) untuk request HTML/CSS/JS.
  - Static file server (webhandler.go) untuk request HTML/CSS/JS.
- Terdapat alur bolak-balik yang Dimana mengirimkan hasil ke client side.



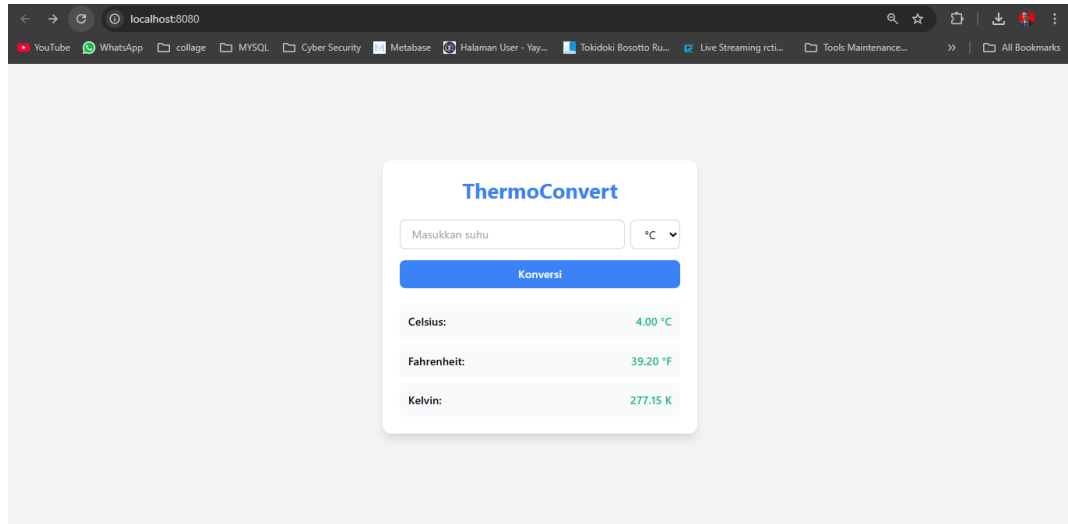
### 3.2. Diagram Alir



### 3.3. Desain Antarmuka

Antarmuka web dirancang dengan prinsip:

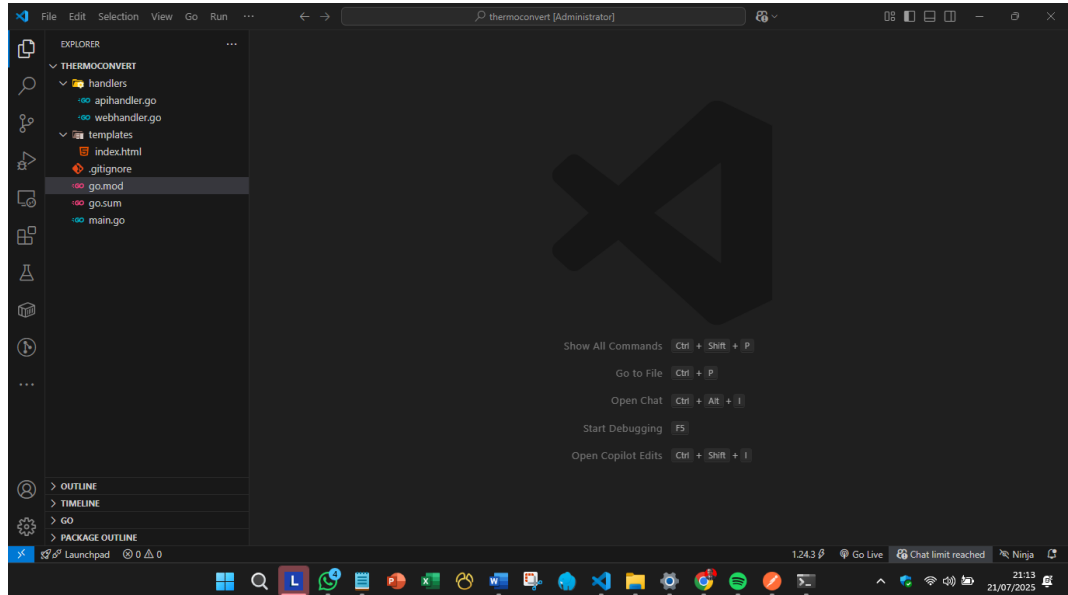
1. **Kesederhanaan:** Form input yang minimalis dengan satu field input
2. **Kemudahan Penggunaan:** Dropdown untuk memilih satuan suhu awal
3. **Keterbacaan:** Hasil konversi ditampilkan dalam bentuk card yang jelas
4. **Responsif:** Desain yang adaptif untuk berbagai ukuran layar



## BAB IV

### IMPLEMENTASI DAN PENGUJIAN

#### 4.1 Struktur Modul/Program



Main.go :entry point aplikasi

Handlers :

apihandler.go : Logika API konversi suhu

webhandler.go : Handler untuk antarmuka web

templates/

index.html : Template HTML antarmuka web

go.mod : File dependensi modul

go.sum : Checksum dependensi

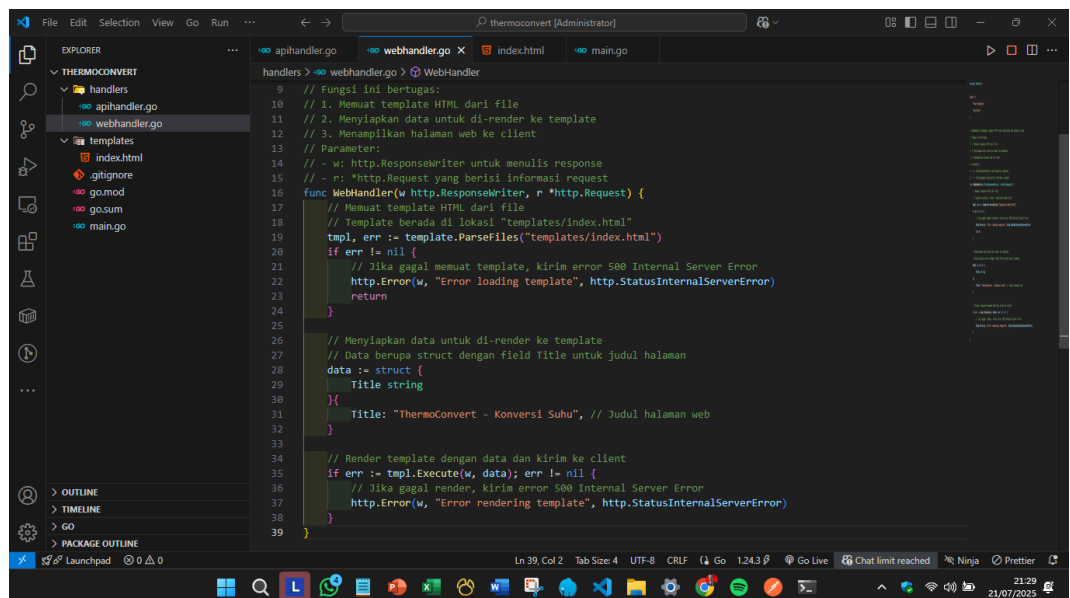
## 4.2 Penggunaan fungsi/metode

1. main.go: Menginisialisasi router dan menghubungkan handler.
2. apihandler.go:  
APIHandler: Menangani logika konversi suhu.  
respondWithJSON: Helper untuk response JSON.  
respondWithError: Helper untuk response error.
3. webhandler.go : Menampilkan antarmuka web

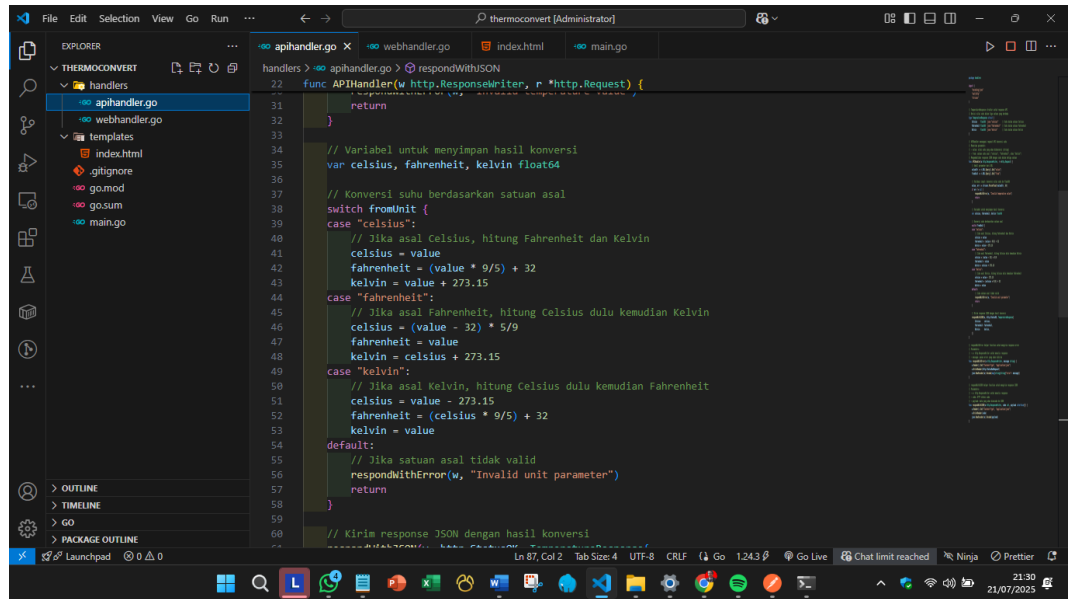
## 4.3 Penanganan Error

Penanganan error dilakukan pada webhandler.go dan apihandler.go dengan kriteria yang penulis jabarkan sebagai berikut :

1. Validasi input numerik
2. Penanganan satuan suhu yang tidak valid
3. Penanganan error template HTML
4. Response HTTP status code yang sesuai



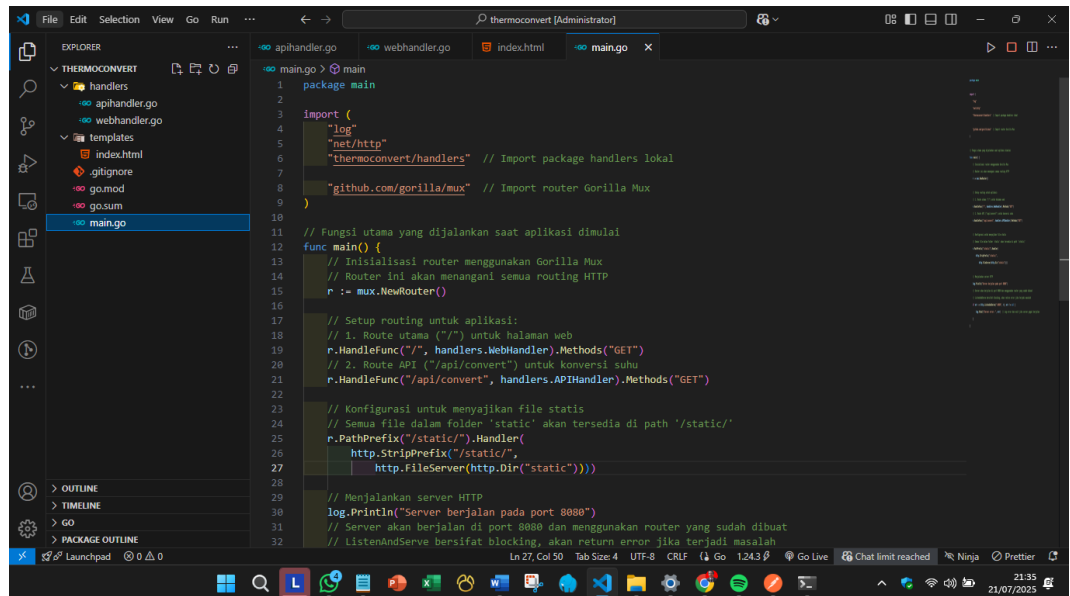
```
handlers > webhandler.go > WebHandler
9 // Fungsi ini bertugas:
10 // 1. Memuat template HTML dari file
11 // 2. Menyiapkan data untuk di-render ke template
12 // 3. Menampilkan halaman web ke client
13 // Parameter:
14 // - w: http.ResponseWriter untuk menulis response
15 // - r: *http.Request yang berisi informasi request
16 func WebHandler(w http.ResponseWriter, r *http.Request) {
17     // Memuat template HTML dari file
18     // Template berada di lokasi "templates/index.html"
19     tpl, err := template.ParseFiles("templates/index.html")
20     if err != nil {
21         // Jika gagal memuat template, kirim error 500 Internal Server Error
22         http.Error(w, "Error loading template", http.StatusInternalServerError)
23         return
24     }
25
26     // Menyiapkan data untuk di-render ke template
27     // Data berupa struct dengan field Title untuk judul halaman
28     data := struct {
29         Title string
30     }{
31         Title: "ThermoConvert - Konversi Suhu", // Judul halaman web
32     }
33
34     // Render template dengan data dan kirim ke client
35     if err := tpl.Execute(w, data); err != nil {
36         // Jika gagal render, kirim error 500 Internal Server Error
37         http.Error(w, "Error rendering template", http.StatusInternalServerError)
38     }
39 }
```



## 4.4 Dokumentasi Internal

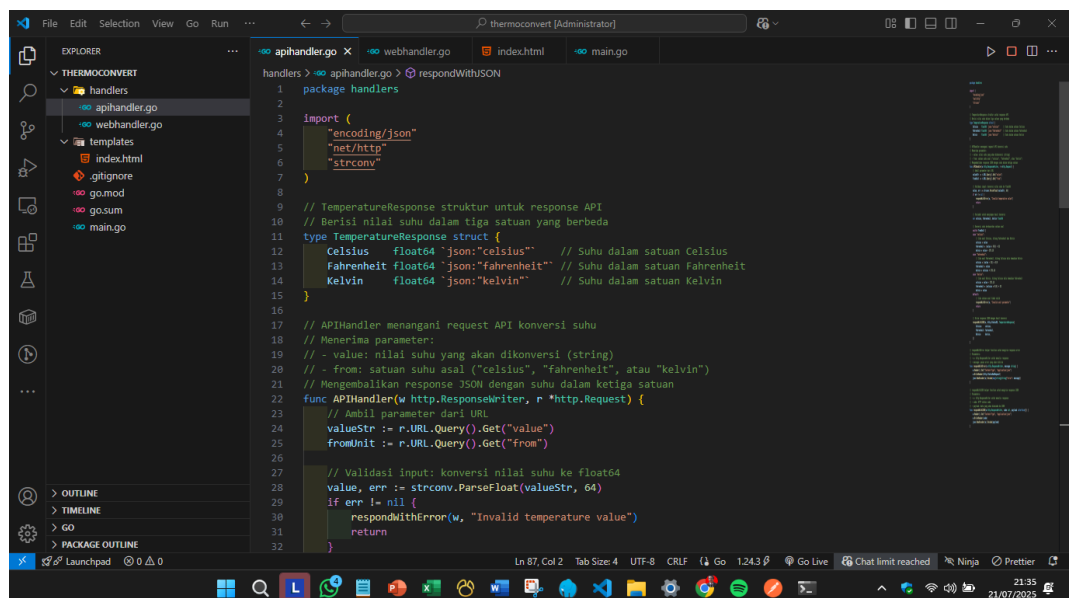
Adapun dokumentasi internal yaitu pemberian komentar untuk menjelaskan setiap logika program, file yang diberi komentar penjelasan adalah :

### 1. Main.go :



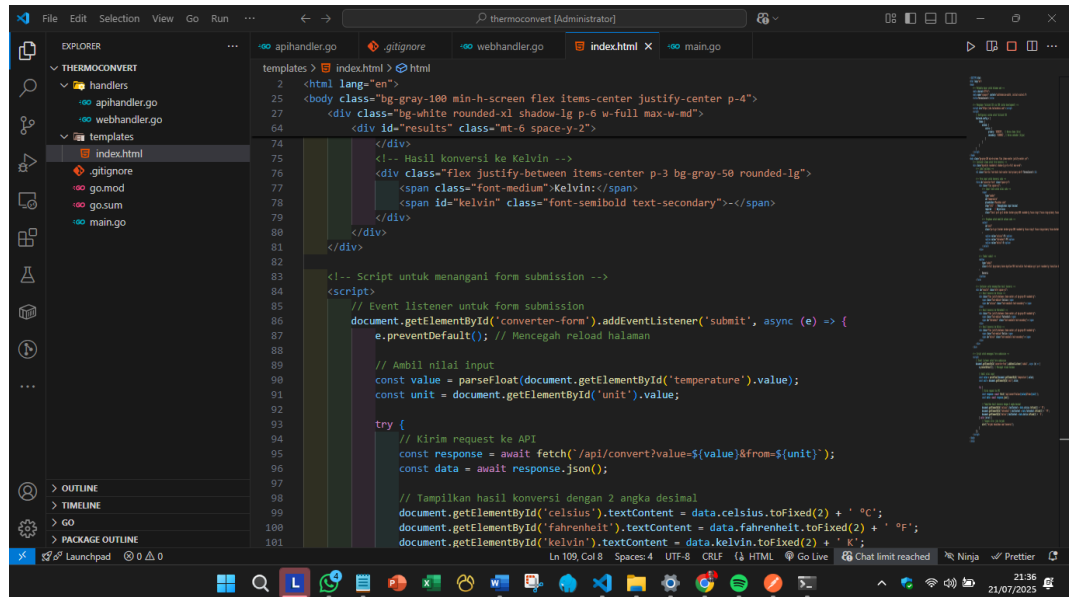
```
1 package main
2
3 import (
4     "log"
5     "net/http"
6     "thermoconvert/handlers" // Import package handlers lokal
7     "github.com/gorilla/mux" // Import router Gorilla Mux
8 )
9
10 // Fungsi utama yang dijalankan saat aplikasi dimulai
11
12 func main() {
13     // Inisialisasi router menggunakan Gorilla Mux
14     // Router ini akan menangani semua routing HTTP
15     r := mux.NewRouter()
16
17     // Setup routing untuk aplikasi:
18     // 1. Route utama ("/") untuk halaman web
19     r.HandleFunc("/", handlers.WebHandler).Methods("GET")
20     // 2. Route API ("/api/convert") untuk konversi suhu
21     r.HandleFunc("/api/convert", handlers.APIHandler).Methods("GET")
22
23     // Konfigurasi untuk menyajikan file statis
24     // Semua file dalam folder 'static' akan tersedia di path '/static/'
25     r.PathPrefix("/static/").Handler(
26         http.StripPrefix("/static/",
27             http.FileServer(http.Dir("static"))))
28
29     // Manjalankan server HTTP
30     log.Println("Server berjalan pada port 8080")
31     // Server akan berjalan di port 8080 dan menggunakan router yang sudah dibuat
32     // ListenAndServe bersifat blocking, akan return error jika terjadi masalah
```

### 2. apihandler.go



```
1 package handlers
2
3 import (
4     "encoding/json"
5     "net/http"
6     "strconv"
7 )
8
9 // TemperatureResponse struktur untuk response API
10 // Berisi nilai suhu dalam tiga satuan yang berbeda
11 type TemperatureResponse struct {
12     celsius float64 `json:"celsius"` // Suhu dalam satuan Celsius
13     fahrenheit float64 `json:"fahrenheit"` // Suhu dalam satuan Fahrenheit
14     kelvin float64 `json:"kelvin"` // Suhu dalam satuan Kelvin
15 }
16
17 // APIHandler menangani request API konversi suhu
18 // Menerima parameter:
19 // - value: nilai suhu yang akan dikonversi (string)
20 // - from: satuan suhu asal ("celsius", "fahrenheit", atau "kelvin")
21 // Mengembalikan response JSON dengan suhu dalam ketiga satuan
22 func APIHandler(w http.ResponseWriter, r *http.Request) {
23     // Ambil parameter dari URL
24     valueStr := r.URL.Query().Get("value")
25     fromUnit := r.URL.Query().Get("from")
26
27     // Validasi input: konversi nilai suhu ke float64
28     value, err := strconv.ParseFloat(valueStr, 64)
29     if err != nil {
30         respondWithError(w, "Invalid temperature value")
31         return
32     }
```

### 3. templates/index.html



The screenshot shows the Visual Studio Code editor interface. On the left, the Explorer sidebar displays the project structure for 'THERMOCONVERT', including files like 'apihandler.go', 'webhandler.go', 'main.go', and the 'templates' directory. The 'templates' directory is expanded, showing 'index.html'. The main editor area displays the content of 'index.html'. The code is a mix of HTML and JavaScript. It includes a basic HTML structure with a body class and a container for results. A JavaScript script is embedded, handling form submissions by fetching data from an API endpoint, parsing the response, and updating the DOM with the converted values for Celsius, Fahrenheit, and Kelvin. The code is well-commented in Indonesian.

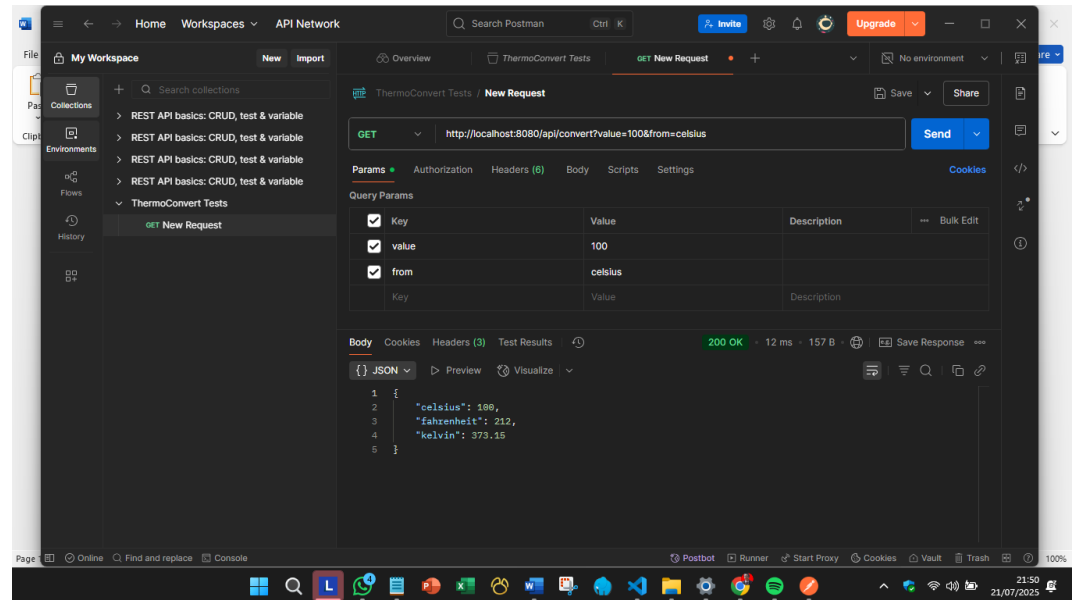
```
1 <!-- index.html -->
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Thermoconvert</title>
7 </head>
8 <body class="bg-gray-100 min-h-screen flex items-center justify-center p-4">
9 <div class="bg-white rounded-xl shadow-lg p-6 w-full max-w-md">
10 <div id="results" class="mt-6 space-y-2">
11 </div>
12 </div>
13 </body>
14 </html>
15
16 <!-- Hasil konversi ke Kelvin -->
17 <div class="flex justify-between items-center p-3 bg-gray-50 rounded-lg">
18 <span class="font-medium">Kelvin:</span>
19 <span id="kelvin" class="font-semibold text-secondary"></span>
20 </div>
21
22 <!-- Script untuk menangani form submission -->
23 <script>
24 // Event listener untuk form submission
25 document.getElementById('converter-form').addEventListener('submit', async (e) => {
26 e.preventDefault(); // Mencegah reload halaman
27
28 // Ambil nilai input
29 const value = parseFloat(document.getElementById('temperature').value);
30 const unit = document.getElementById('unit').value;
31
32 try {
33 // Kirim request ke API
34 const response = await fetch(`/api/convert?value=${value}&from=${unit}`);
35 const data = await response.json();
36
37 // Tampilkan hasil konversi dengan 2 angka desimal
38 document.getElementById('celsius').textContent = data.celsius.toFixed(2) + ' °C';
39 document.getElementById('fahrenheit').textContent = data.fahrenheit.toFixed(2) + ' °F';
40 document.getElementById('kelvin').textContent = data.kelvin.toFixed(2) + ' K';
41 } catch (error) {
42 console.error('Error: ', error);
43 }
44 }
45
46 </script>
```

## 4.5 Pengujian

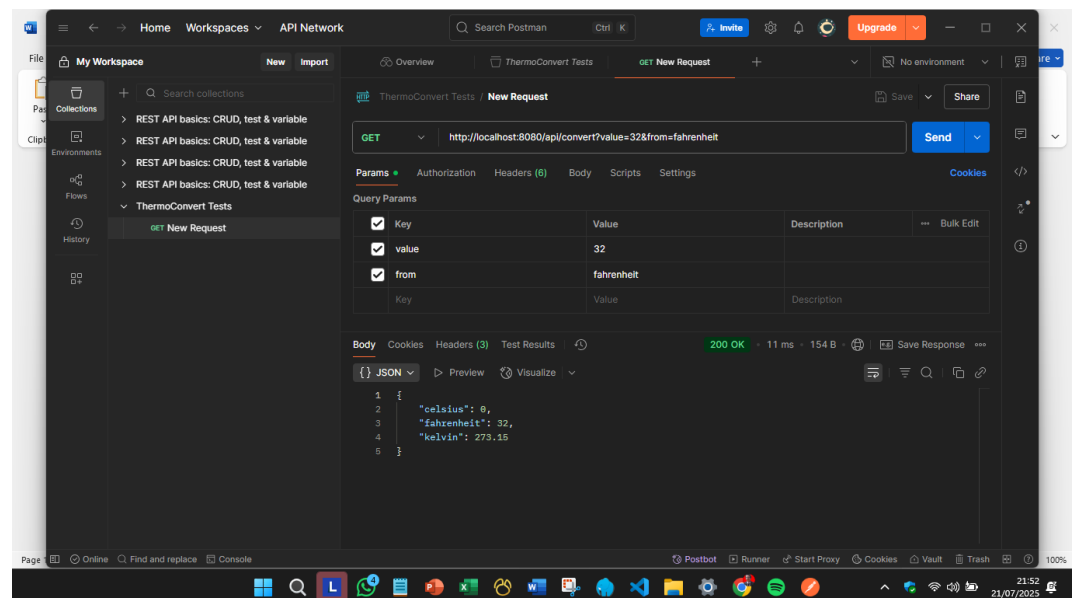
Adapun pengujian dilakukan melalui interface dan testing lewat *API*:

Pengujian via API :

### 1. Konversi dari Celsius

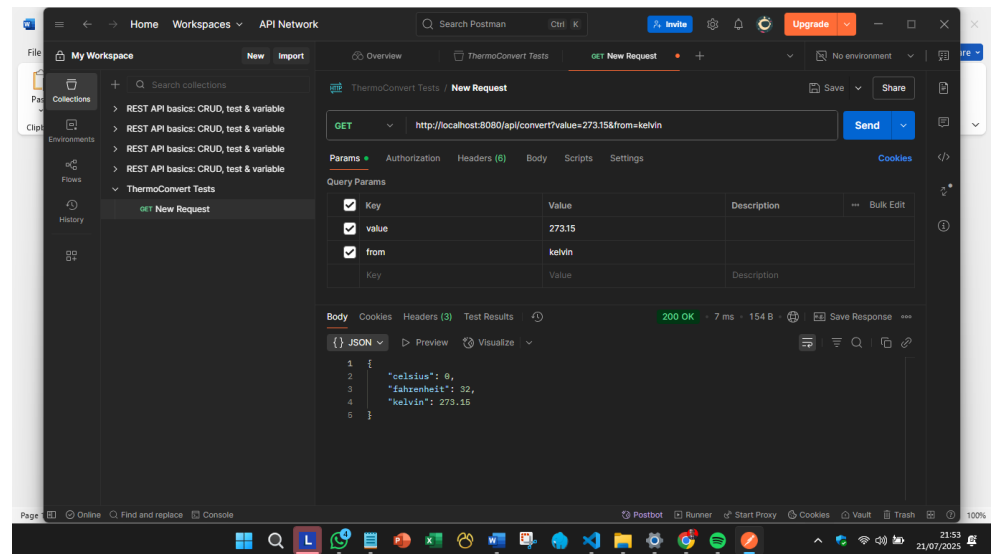


### 2. Konversi dari Fahrenheit

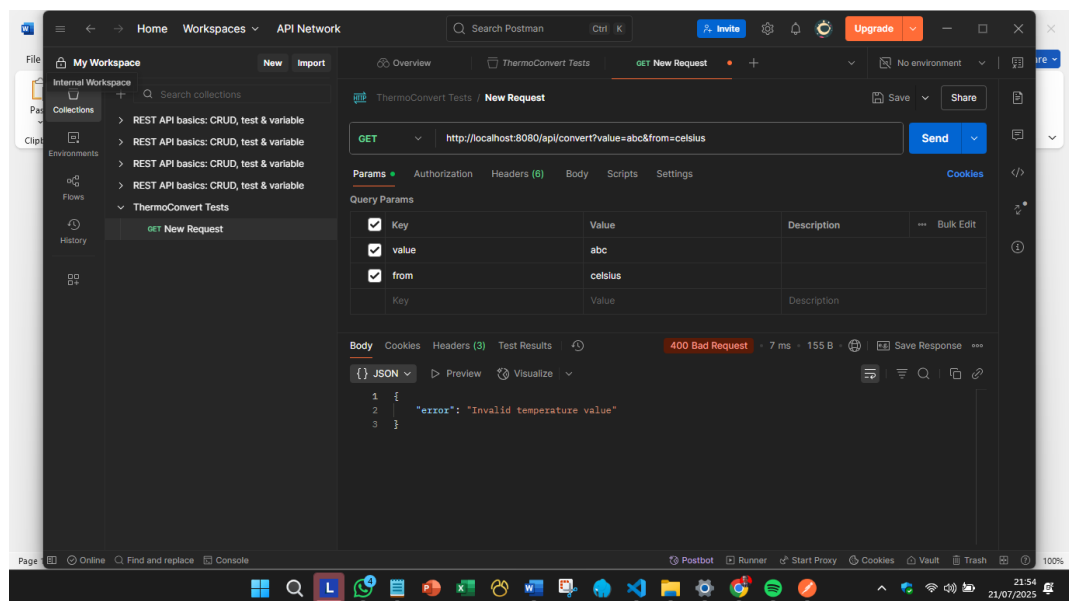




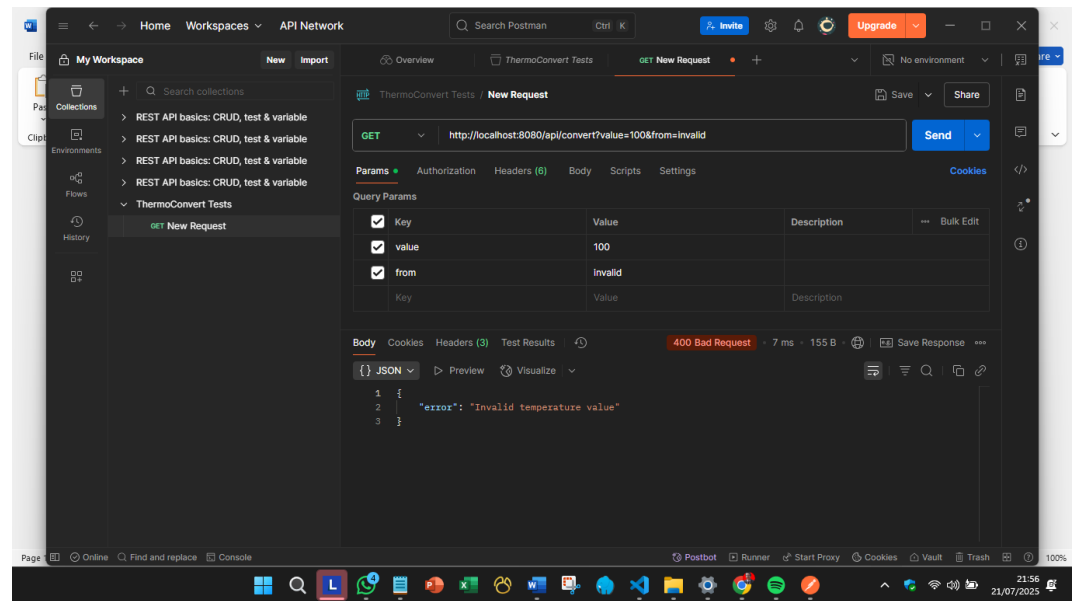
### 3. Konversi dari kelvin



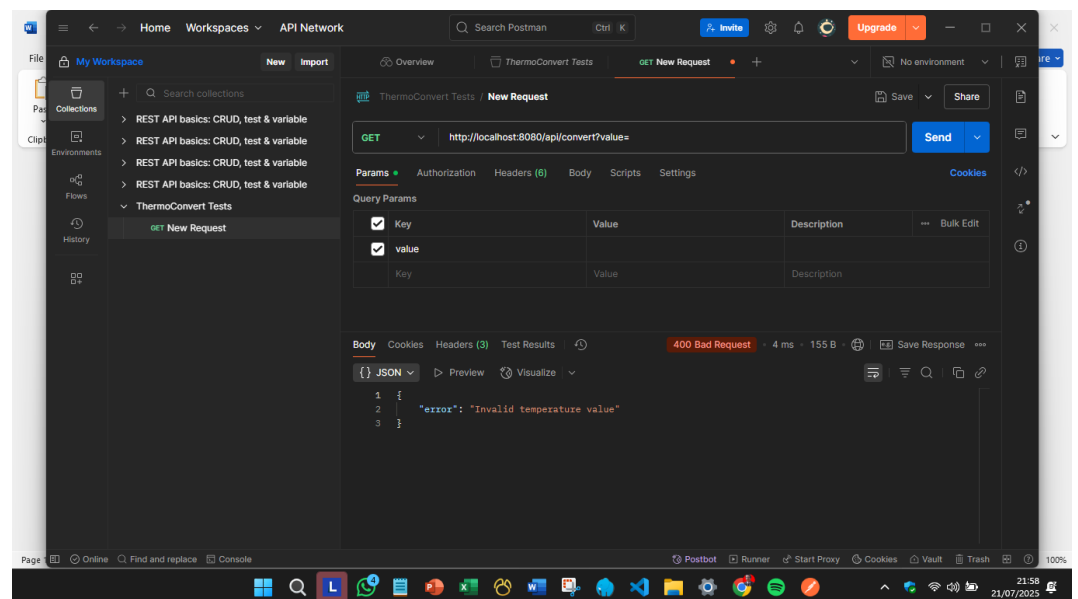
### 4. Konversi ke Celsius jika value bukan angka



## 5. Konversi jika satuan suhu invalid

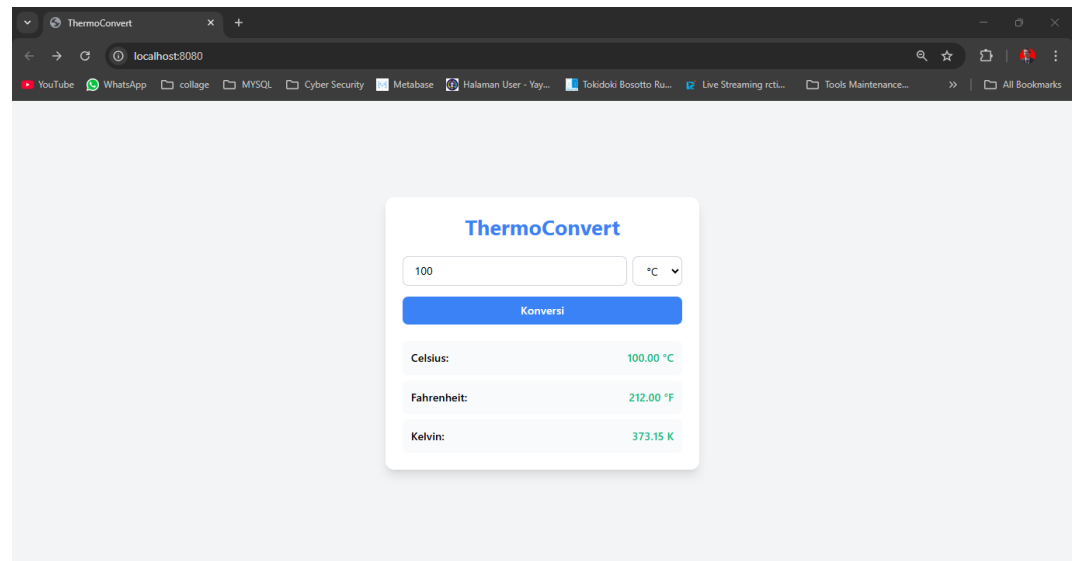


## 6. jika satuan suhu dan value kosong

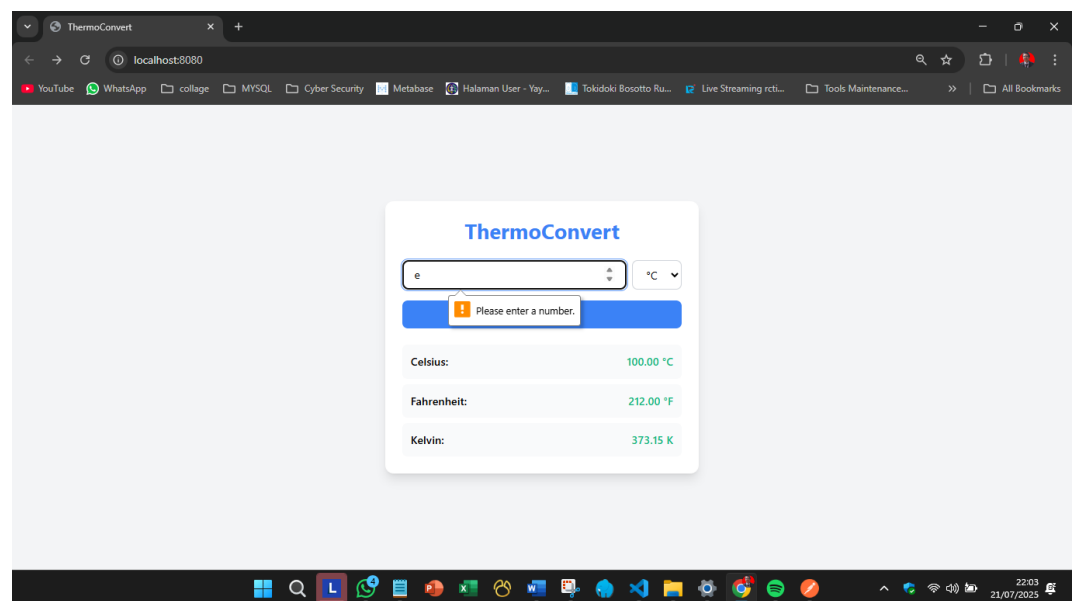


Adapun pengujian via interface adalah sebagai berikut :

### 1. Konversi berupa angka



### 2. Pengujian jika teks



[illegible]

## **BAB V**

### **SARAN DAN KESIMPULAN**

#### **5.1 Saran**

Berdasarkan hasil pengujian dan implementasi aplikasi ThermoConvert, berikut beberapa saran untuk pengembangan lebih lanjut:

1. Peningkatan Fitur:
  - Menambahkan konversi satuan suhu lainnya (Reamur, Rankine).
  - Menyimpan riwayat konversi pengguna (local storage/database).
  - Menambahkan visualisasi grafik perubahan suhu.
2. Optimasi Teknis:
  - Implementasi caching untuk mengurangi beban server pada request yang sama.
  - Migrasi ke framework modern seperti Gin atau Echo untuk performa lebih baik.
  - Penambahan unit testing yang lebih komprehensif (benchmark, load testing).
3. Keamanan:
  - Penambahan rate limiting untuk mencegah abuse API.
  - Implementasi HTTPS jika digunakan di production.
  - Sanitasi input untuk mencegah serangan XSS/SQL Injection.

## 5.2 Kesimpulan

Aplikasi ThermoConvert telah berhasil dibangun sebagai solusi konversi suhu berbasis web dan API menggunakan Golang. Berdasarkan pengujian yang dilakukan:

### 1. Fungsionalitas:

- Aplikasi dapat mengkonversi suhu (Celsius, Fahrenheit, Kelvin) dengan akurat.
- API bekerja sesuai ekspektasi dan mampu menangani error input dengan baik.

### 2. Kinerja:

- Respons API cepat (<100ms) untuk konversi tunggal.
- Antarmuka web responsif di berbagai perangkat.

### 3. Keterbatasan:

- Belum mendukung konversi batch (multi-input sekaligus).
- Tidak ada autentikasi untuk penggunaan API.

## DAFTAR PUSTAKA

Donovan, A. A. A., & Kernighan, B. W. (2021). *\*The Go Programming Language\** (2nd ed.). Addison-Wesley Professional.

Gorilla Web Toolkit. (2023). *\*Mux Documentation\**.  
<https://www.gorillatoolkit.org/pkg/mux>

Ruangguru. (n.d.). **Suhu dan Kalor**. <https://www.ruangguru.com/blog/suhu>

Shandy Rahmawan. (2015). **Sistem Informasi Manajemen**. Repository BSI.  
[https://repository.bsi.ac.id/index.php/unduh/item/2384/Shandy-Rahmawan\\_13140734.pdf](https://repository.bsi.ac.id/index.php/unduh/item/2384/Shandy-Rahmawan_13140734.pdf)

Tailwind Labs. (2023). *\*Tailwind CSS Documentation\**.  
<https://tailwindcss.com/docs>

## **LAMPIRAN**

<https://github.com/StevanusAndika/thermoconvert>