**LAPORAN PRAKTIKUM LATIHAN**

Pemrograman Berbasis Web

Pertemuan 07



Oleh :

NAMA : FARAH TRI MAHARDINI

NPM : 4522210042

Dosen:

**Adi Wahyu Pribadi ,S.Si.,M.Kom**

**S1-Teknik Informatika**

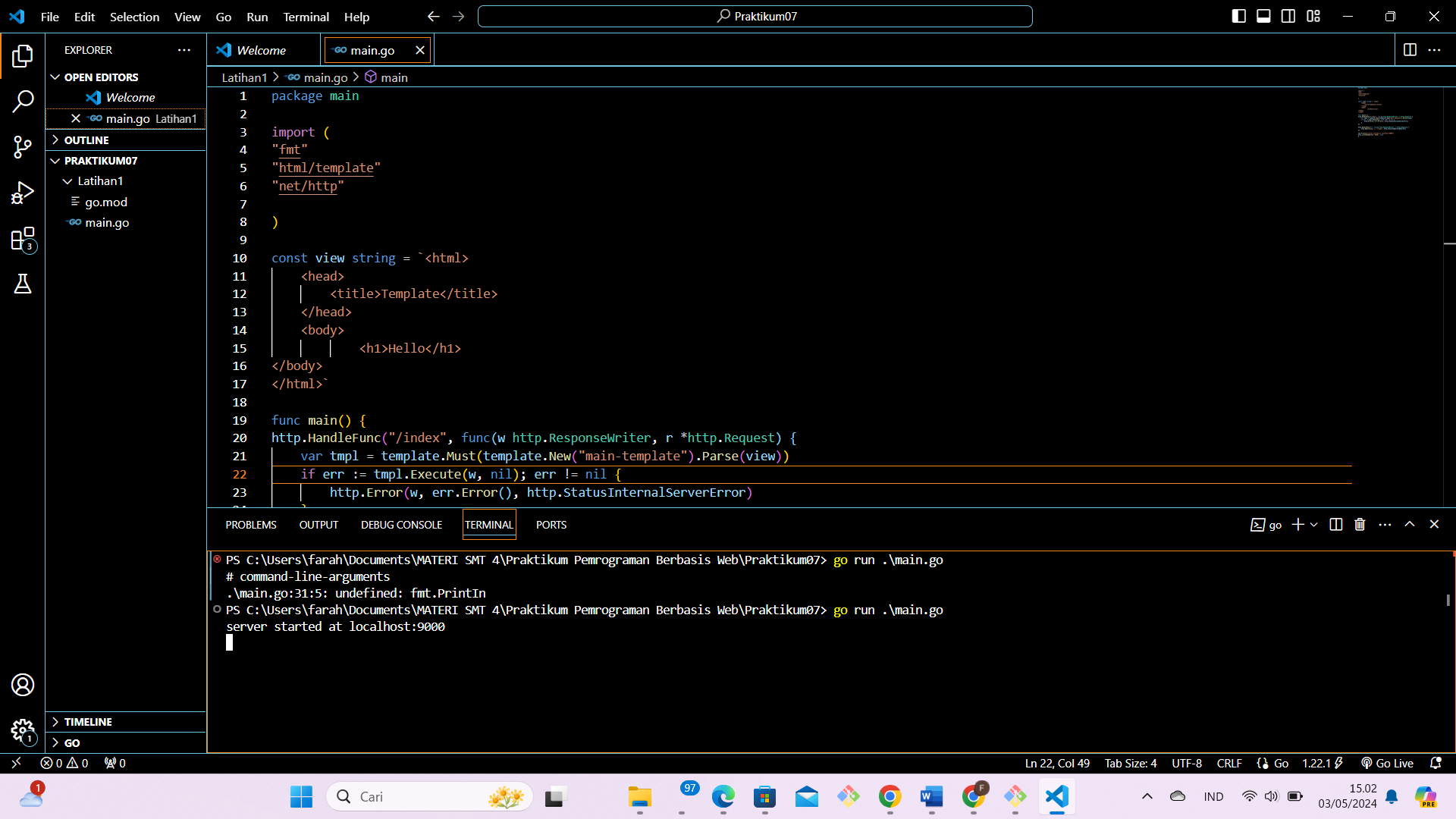
**Fakultas Teknik Universitas Pancasila**

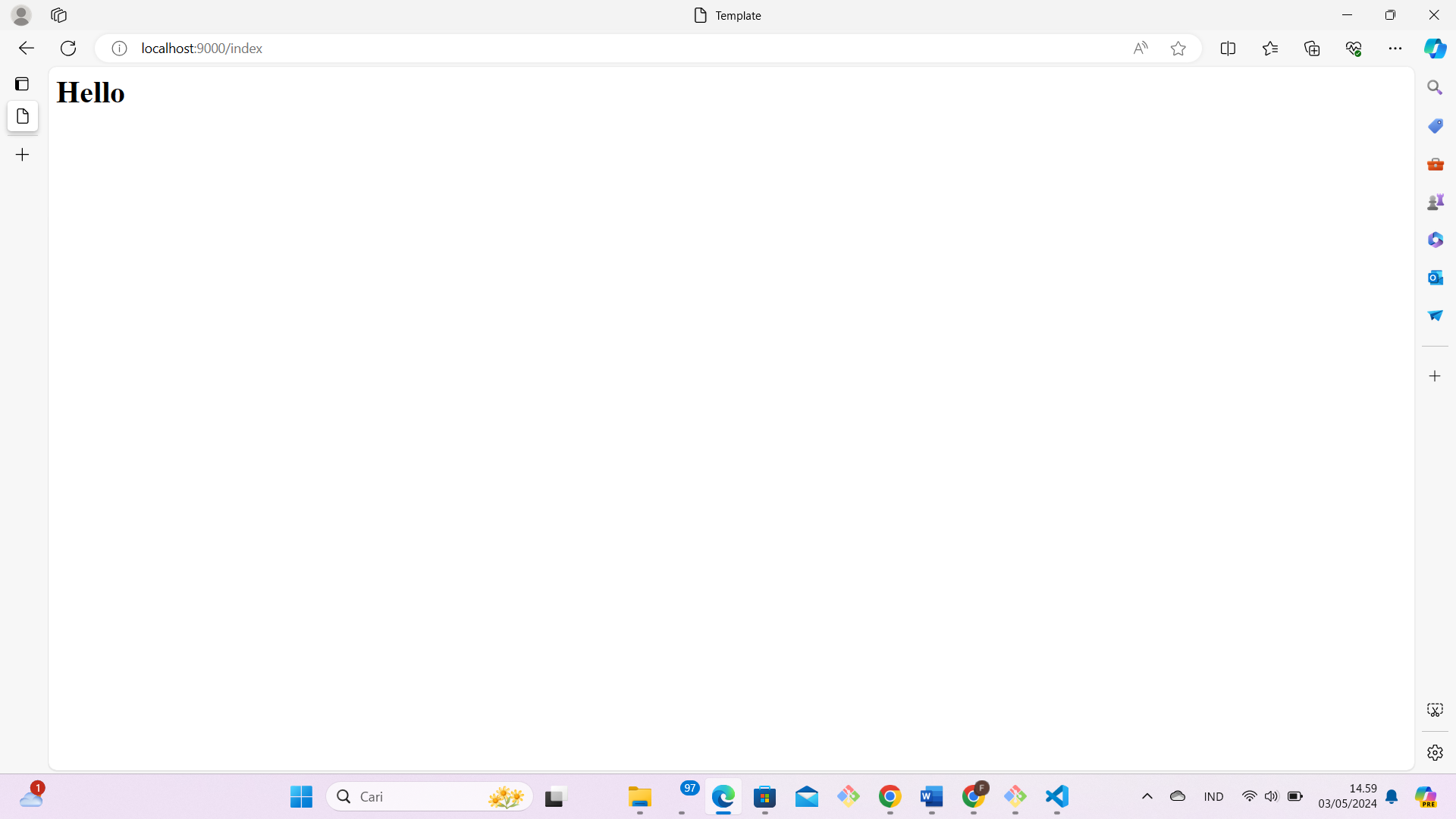
**2023/2024**

Latihan 1 **TEMPLATE: RENDER HTML STRING**

HTML atau (HyperText Markup Language) adalah bahasa markup yang digunakan untuk membuat halaman web. HTML terdiri dari tag yang digunakan untuk menentukan struktur dan konten halaman web. Tag HTML dapat dipisahkan dengan teks biasa.

Parsing Method adalah proses membaca dan memahami teks HTML untuk membangun struktur dokumen yang sesuai. Parsing method biasanya digunakan oleh browser web untuk mengubah teks HTML menjadi halaman web yang dapat dilihat.

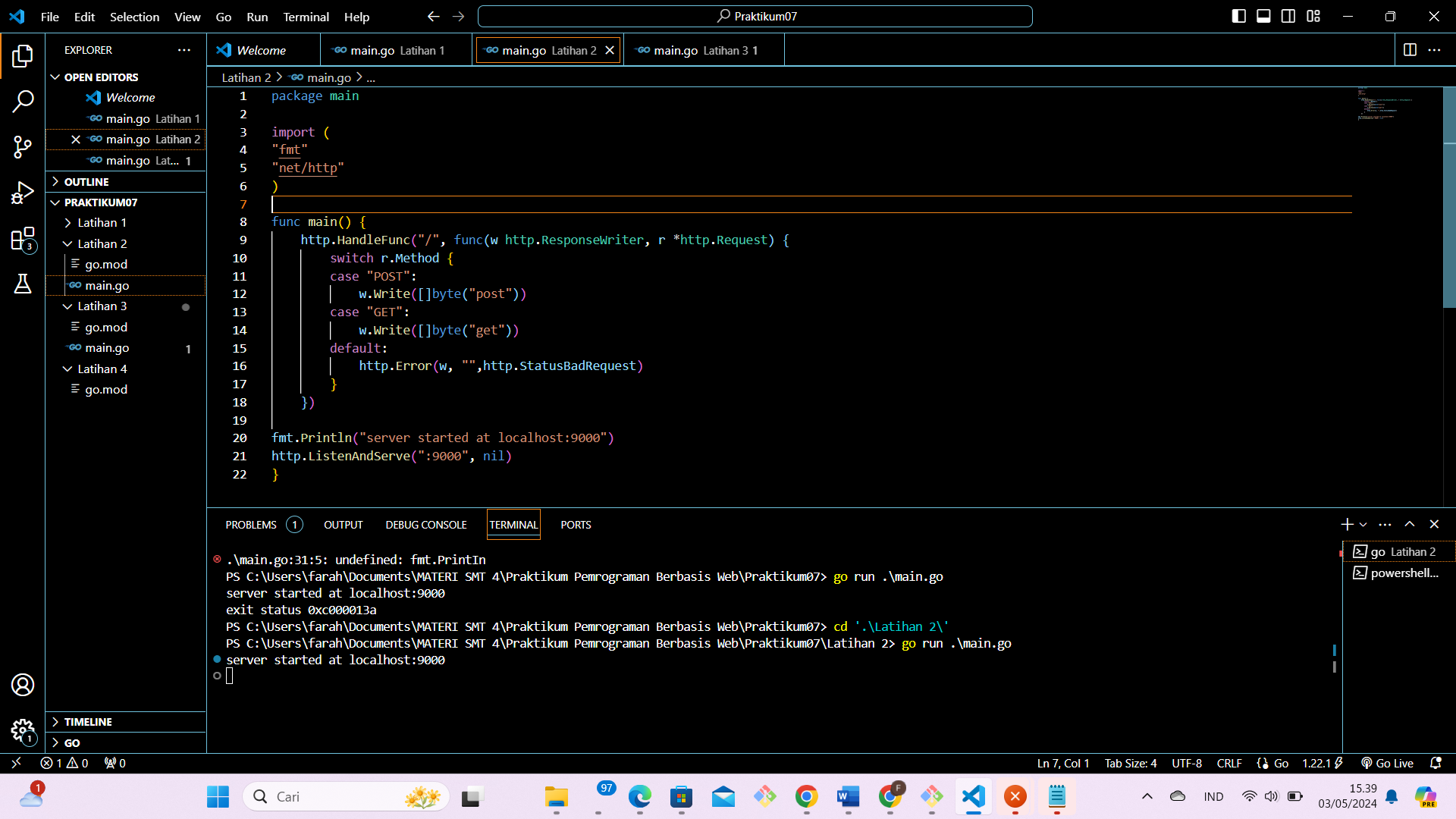




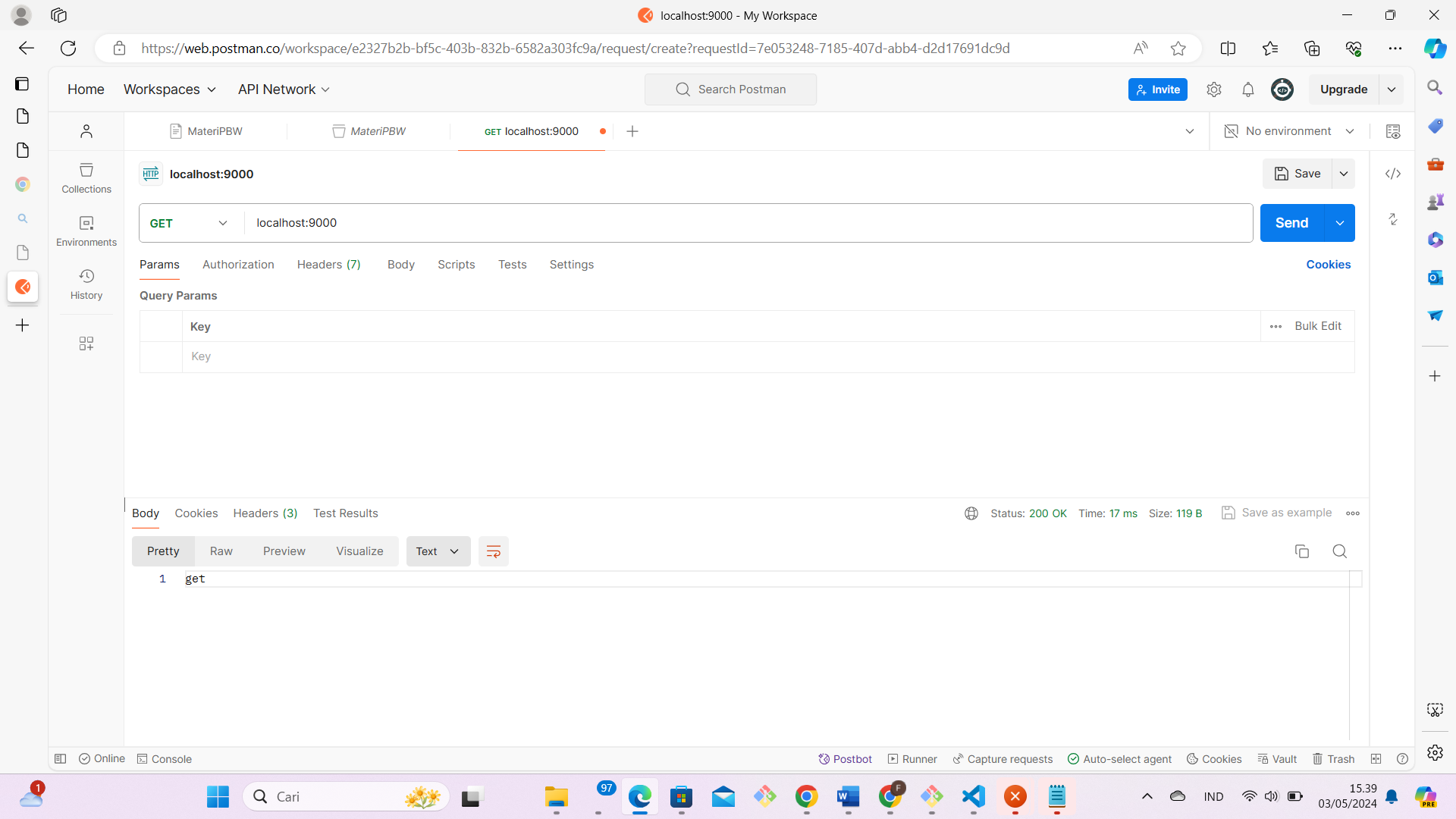
Latihan 2 **HTTP METHOD POST & GET**

Postman adalah sebuah aplikasi yang digunakan untuk menguji performa dan stabilitas sebuah API (Application Programming Interface).

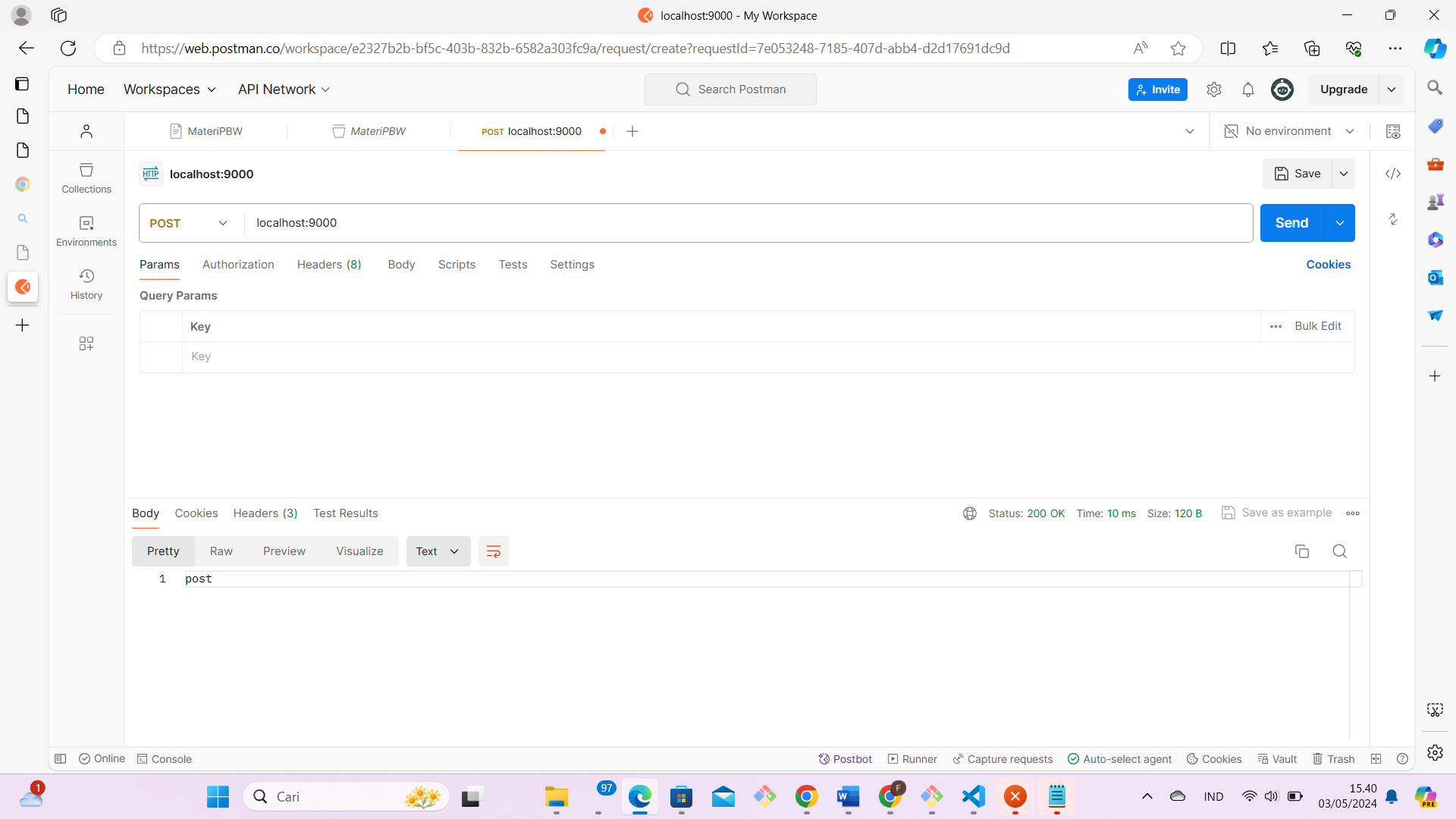
Keguanaan nya Aplikasi ini memungkinkan pengguna untuk mengirimkan permintaan API ke server web dan menerima respons, yang sangat berguna dalam pengembangan aplikasi yang memerlukan integrasi dengan API. Postman juga dapat digunakan untuk menguji API yang telah dibuat, memantau kinerja, dan memodifikasi API untuk meningkatkan kualitas aplikasi yang dipengaruhi oleh API tersebut.



**Method Get**

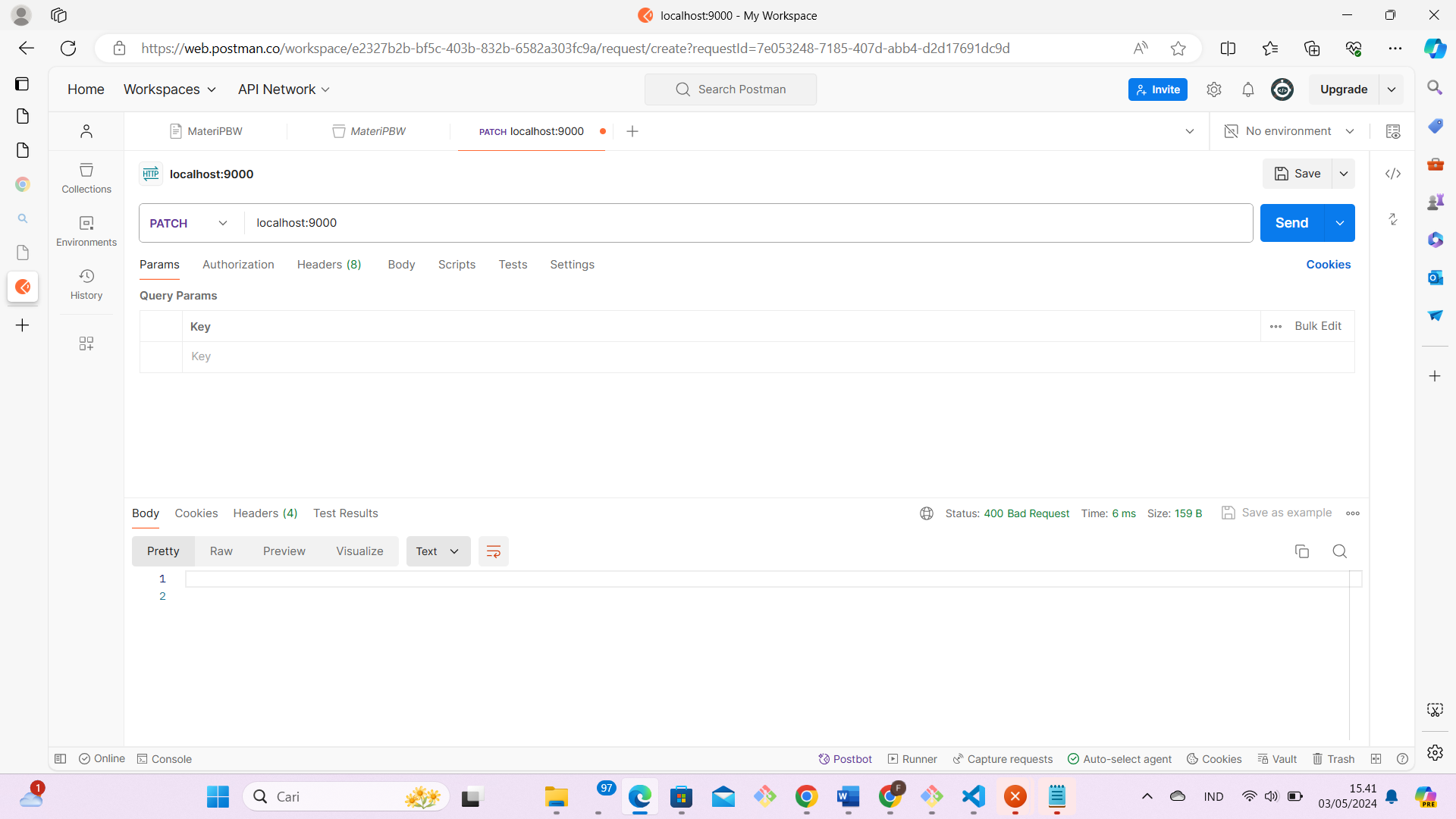


**Method Post**



Menggunakan selain Get dan Post di sini saya menggunakan **method PATCH** menghasilkan 400 Bad Request

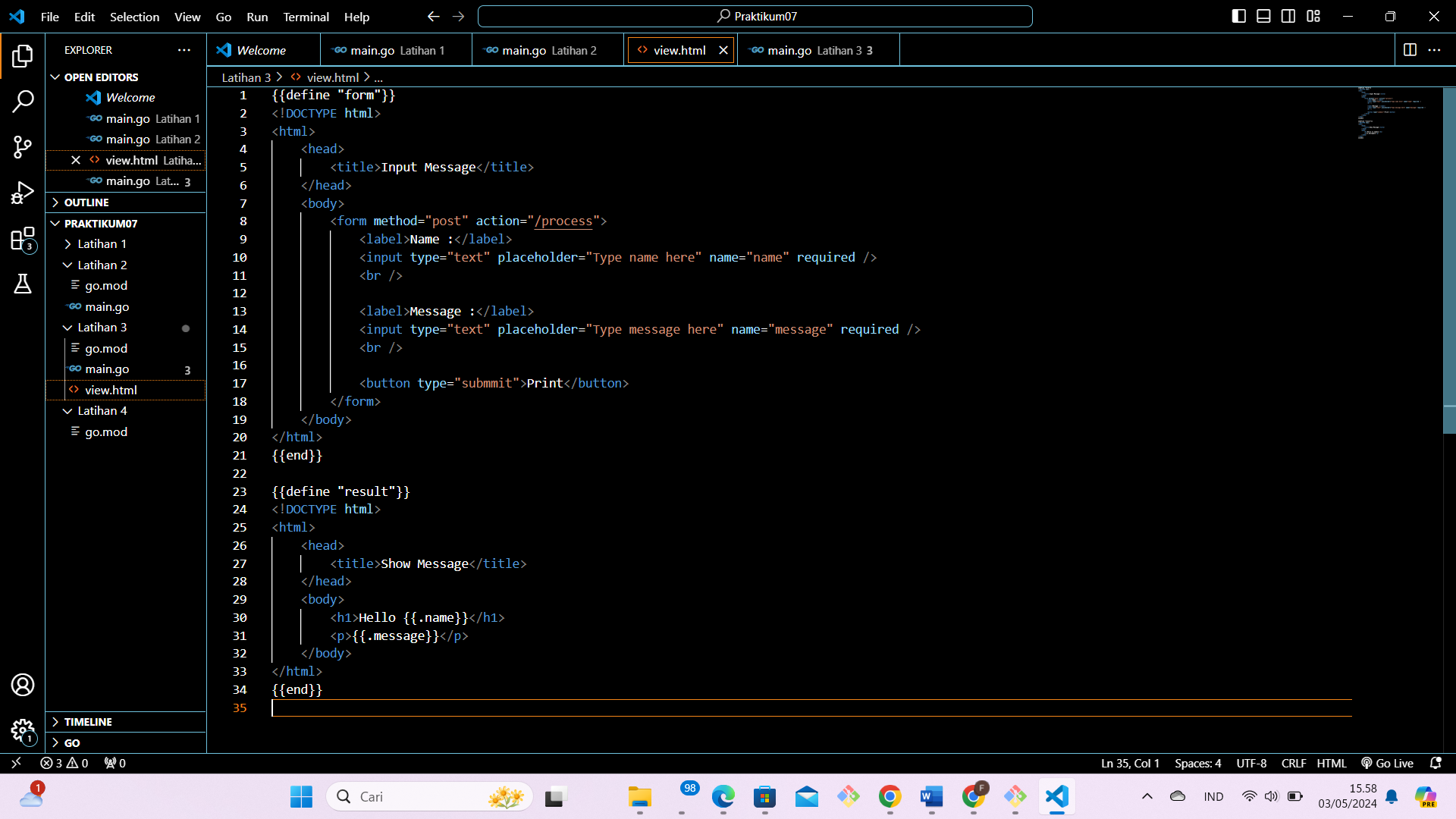
= Karena menunjukkan bahwa server tidak dapat memahami atau memproses permintaan Anda karena sintaks salah atau data yang tidak valid



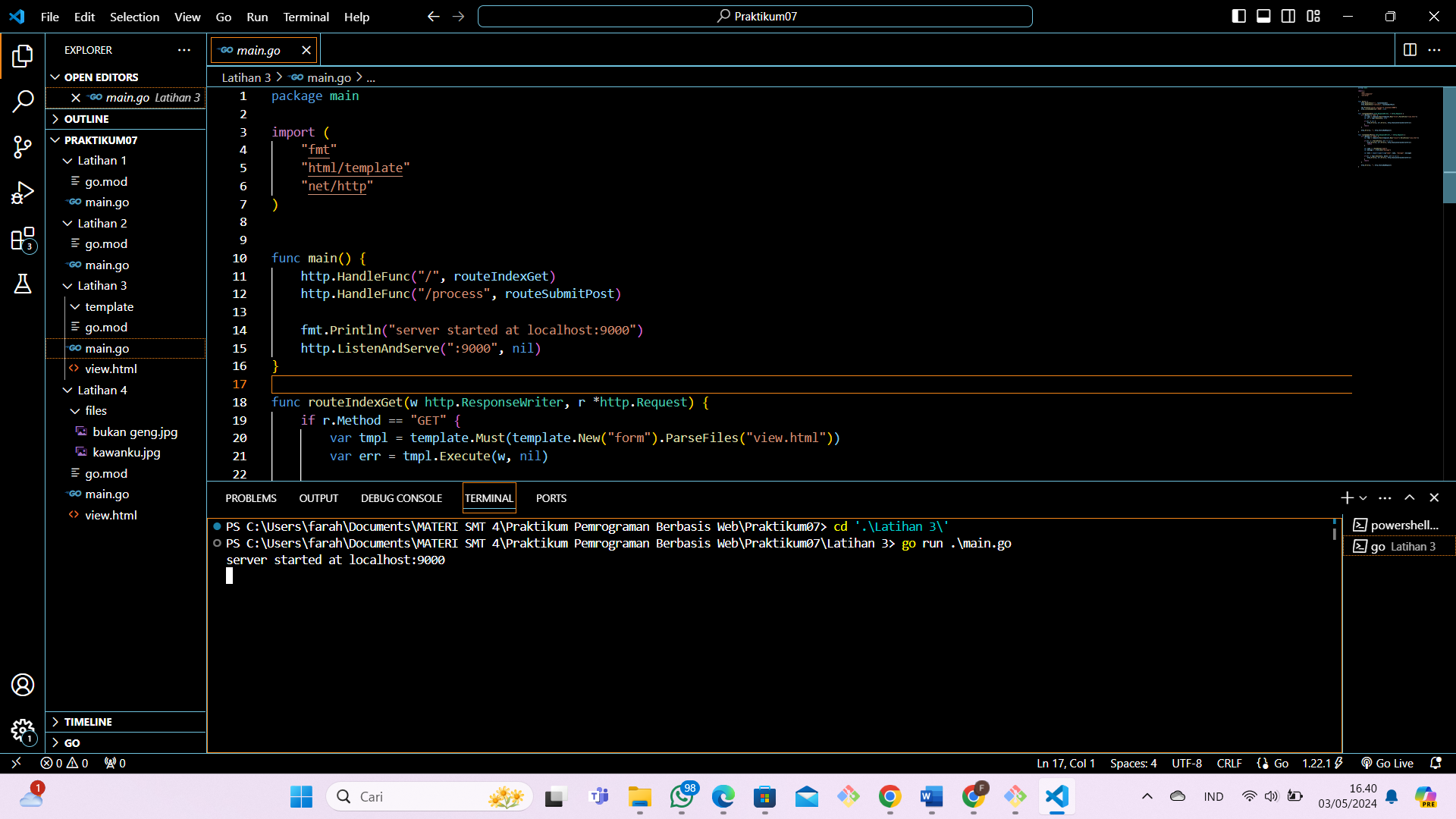
Lantihan 3 **FORM VALUE**

* **Front End** Bayangkan front end sebagai tampilan depan dari sebuah website atau aplikasi. Ini adalah bagian yang Anda lihat dan interaksikan sebagai pengguna.
* **Back End,** terdapat back end yang bekerja keras untuk membuat semuanya berjalan. Back end menangani semua logika, proses, dan komunikasi dengan database.
* **API (Apllication Programming Interface) =** API bertindak sebagai jembatan antara front end dan back end. Ini adalah kumpulan aturan dan spesifikasi yang memungkinkan front end untuk berkomunikasi dengan back end dan mengambil data yang dibutuhkan untuk menampilkan konten atau melakukan tindakan.

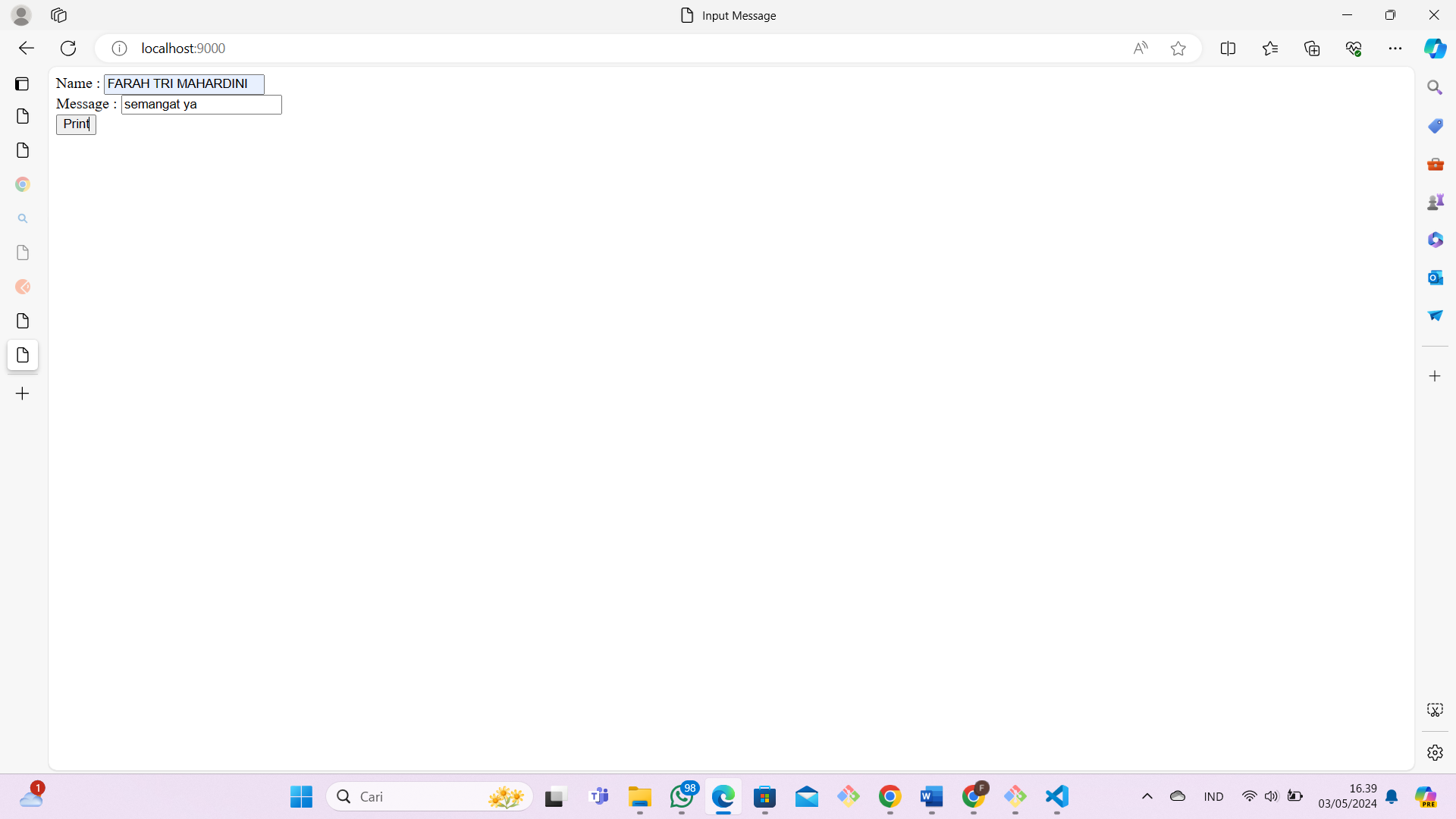
**View.html** bersisi 2 template yaitu form dan result

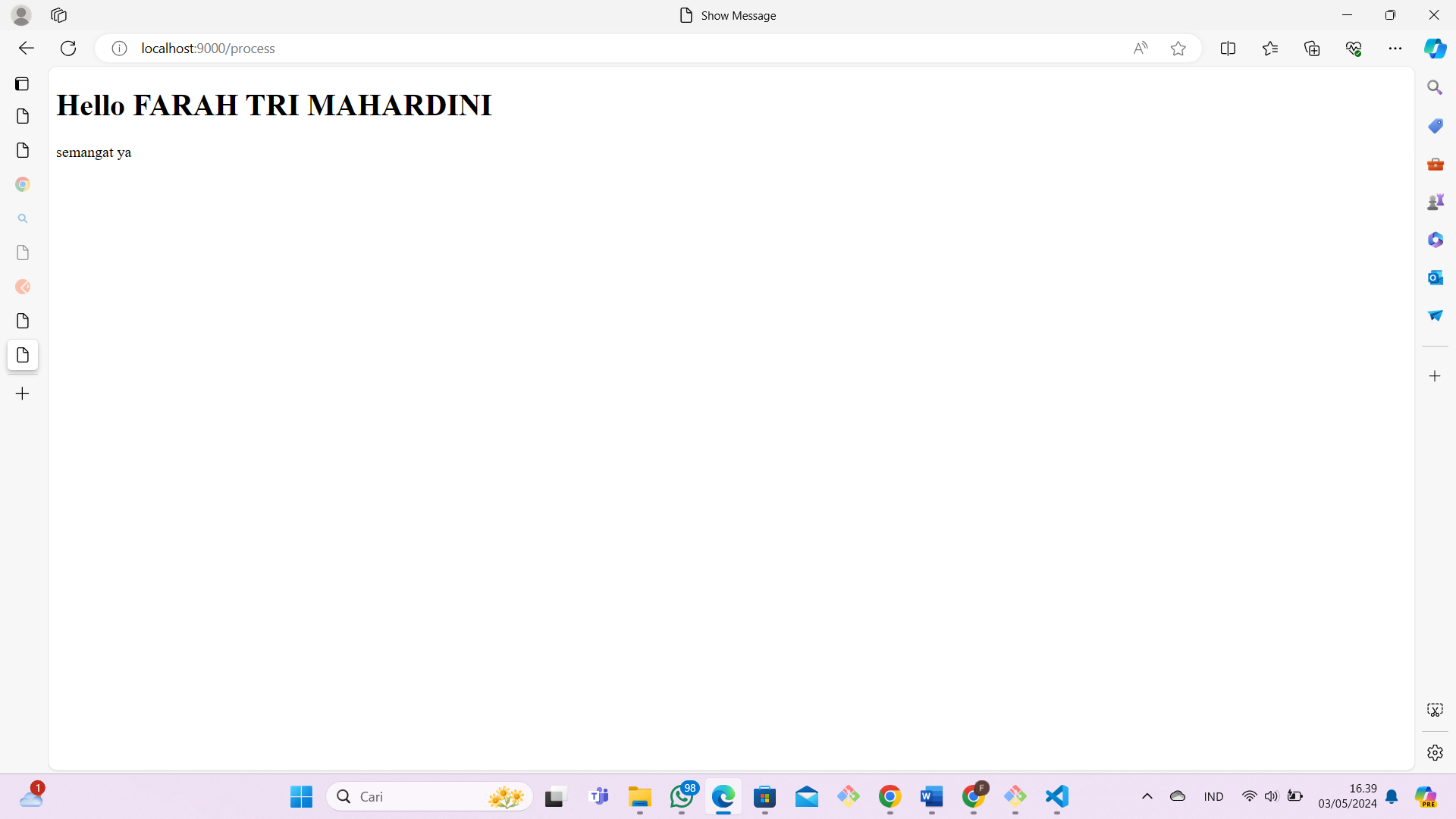


**Main.go**



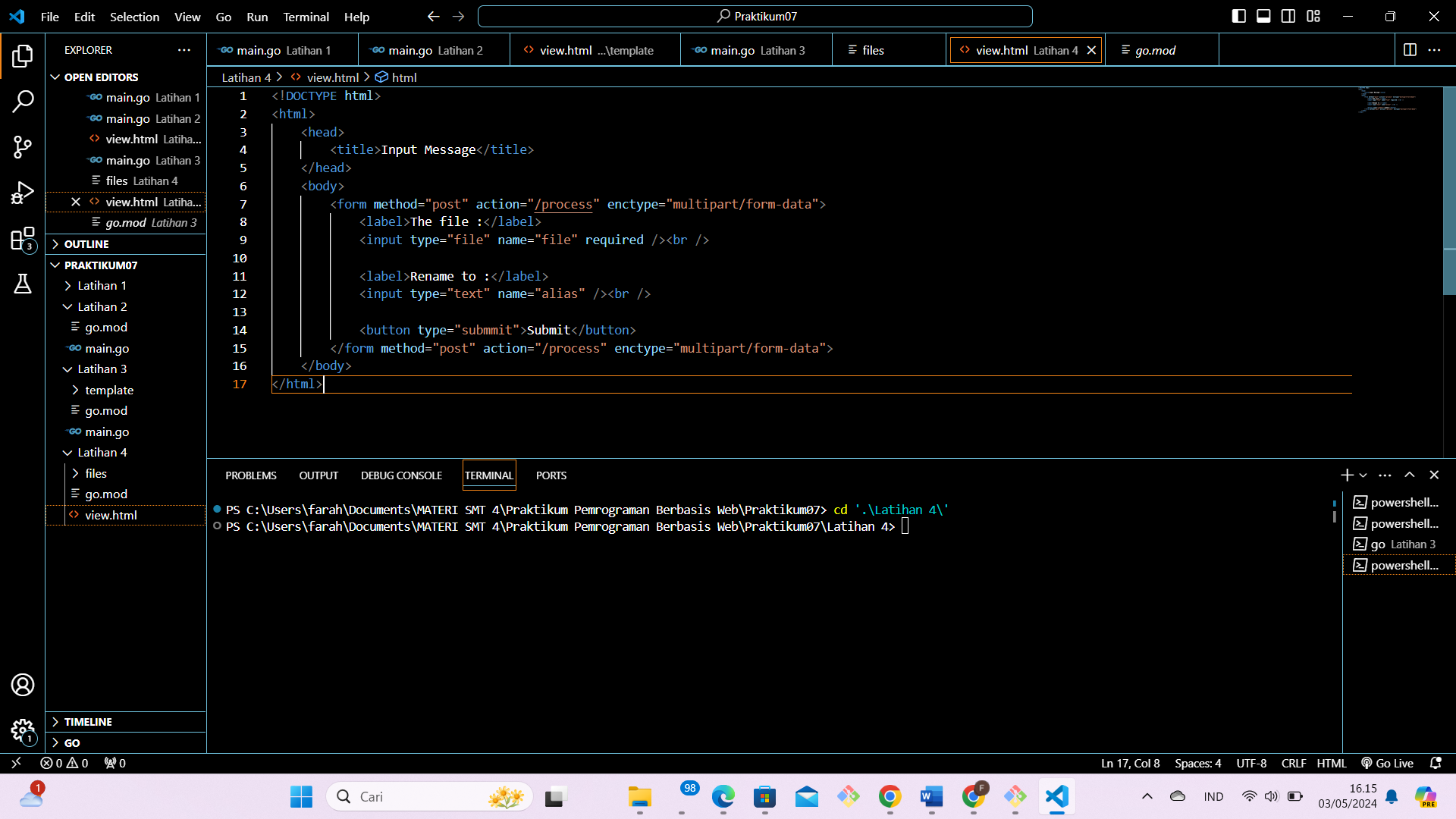
**Testing**





Latihan 4 **FORM UPLOAD FILE**

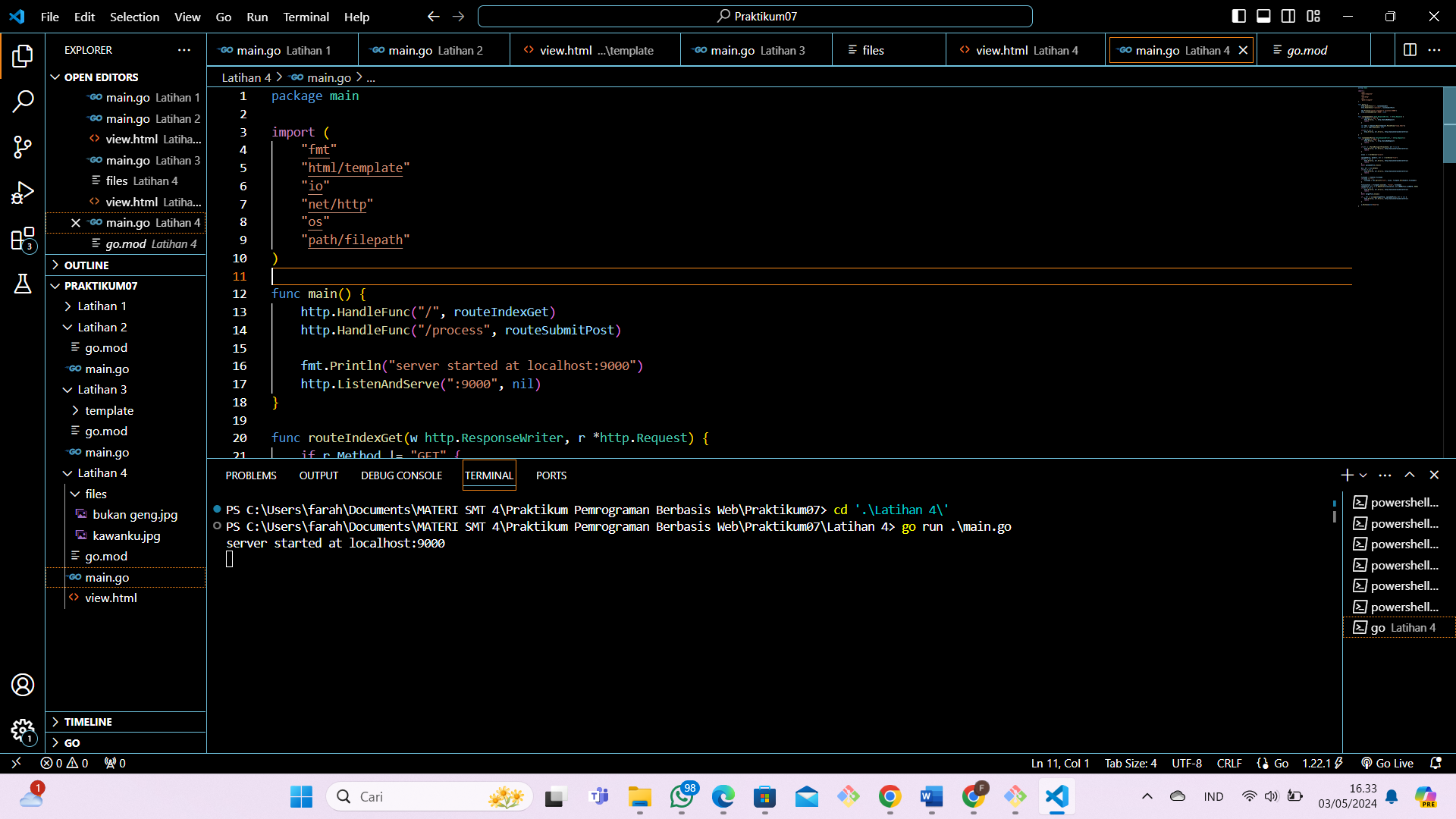
**View.html**



Perlu diperhatikan, pada tag <form> perlu ditambahkan atribut enctype="multipart/form-data”

= Karena Agar form akan dapat meng-handle pengiriman data file

Main.go



**Source Code main.go**

package main

import (

    "fmt"

    "html/template"

    "io"

    "net/http"

    "os"

    "path/filepath"

)

func main() {

    http.HandleFunc("/", routeIndexGet)

    http.HandleFunc("/process", routeSubmitPost)

    fmt.Println("server started at localhost:9000")

    http.ListenAndServe(":9000", nil)

}

func routeIndexGet(w http.ResponseWriter, r \*http.Request) {

    if r.Method != "GET" {

        http.Error(w, "", http.StatusBadRequest)

        return

    }

    var tmpl = template.Must(template.ParseFiles("view.html"))

    var err = tmpl.Execute(w, nil)

    if err != nil {

        http.Error(w, err.Error(), http.StatusInternalServerError)

    }

}

func routeSubmitPost(w http.ResponseWriter, r \*http.Request) {

    if r.Method != "POST" {

        http.Error(w, "", http.StatusBadRequest)

        return

    }

    if err := r.ParseMultipartForm(1024); err != nil {

        http.Error(w, err.Error(), http.StatusInternalServerError)

        return

    }

    alias := r.FormValue("alias")

    uploadedFile, handler, err := r.FormFile("file")

    if err != nil {

        http.Error(w, err.Error(), http.StatusInternalServerError)

        return

    }

    defer uploadedFile.Close()

    dir, err := os.Getwd()

    if err != nil {

        http.Error(w, err.Error(), http.StatusInternalServerError)

        return

    }

    filename := handler.Filename

    if alias != "" {

        filename = fmt.Sprintf("%s%s", alias, filepath.Ext(handler.Filename))

    }

    fileLocation := filepath.Join(dir, "files", filename)

    targetFile, err := os.OpenFile(fileLocation, os.O\_WRONLY|os.O\_CREATE, 0666)

    if err != nil {

        http.Error(w, err.Error(), http.StatusInternalServerError)

        return

    }

    defer targetFile.Close()

    if \_, err := io.Copy(targetFile, uploadedFile); err != nil {

        http.Error(w, err.Error(), http.StatusInternalServerError)

        return

    }

    w.Write([]byte("done"))

}

**Testing**

