

**LAPORAN PRAKTIKUM-07**  
**PEMROGRAMAN BERBASIS WEB**

**KELAS A**

Disusun Untuk Memenuhi Tugas Mata Kuliah Pemrograman Berbasis Web



Disusun oleh:

Gibran Kahlil

4522210128

Dosen Pengampu:

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

**Program Studi Teknik Informatika**  
**Fakultas Teknik Universitas Pancasila**

**2023/2024**

Link GitHub: <https://github.com/gibrankahlil260404/PBW>

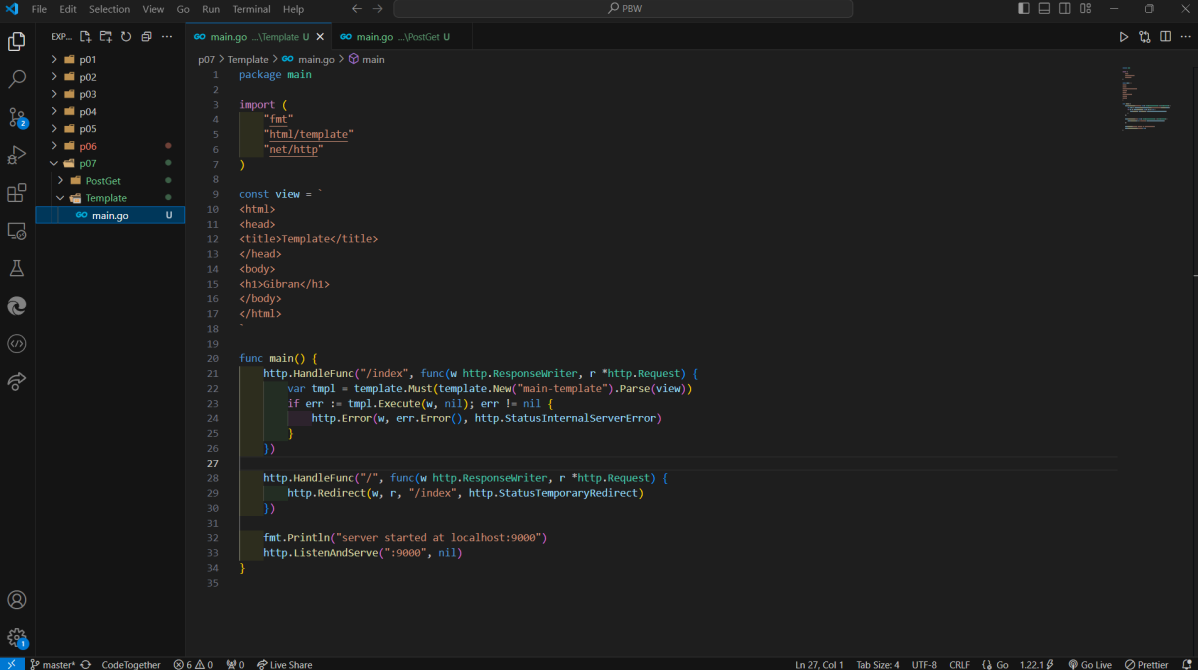
## 1. TEMPLATE: RENDER HTML STRING

- Penjelasan

HTML (Hypertext Markup Language) adalah bahasa markah yang digunakan untuk membuat halaman web. Dengan menggunakan HTML, pengembang dapat mengatur struktur halaman dan menambahkan konten, seperti teks, gambar, tautan, dan formulir. HTML juga mendukung penggunaan atribut untuk memberikan informasi tambahan tentang elemen, seperti warna, ukuran, dan HTML adalah bahasa dasar pengembangan web yang memungkinkan pengembang membuat situs web yang efektif dan menarik.

Metode parsing adalah proses mengurai data dalam format tertentu menjadi struktur data yang dapat dimanipulasi oleh program. Ini biasanya digunakan dalam pengembangan web untuk mengurai atau membaca dokumen HTML atau XML, sehingga program dapat memahami struktur dan konten dokumen tersebut dan dapat mengekstrak atau mengubah data sesuai kebutuhan. Tergantung pada kompleksitas dokumen dan kebutuhan program, berbagai metode parsing dapat digunakan, seperti DOM (Document Object Model), SAX (Simple API for XML), XPath (XML Path Language), dan Regex (Regular Expressions).

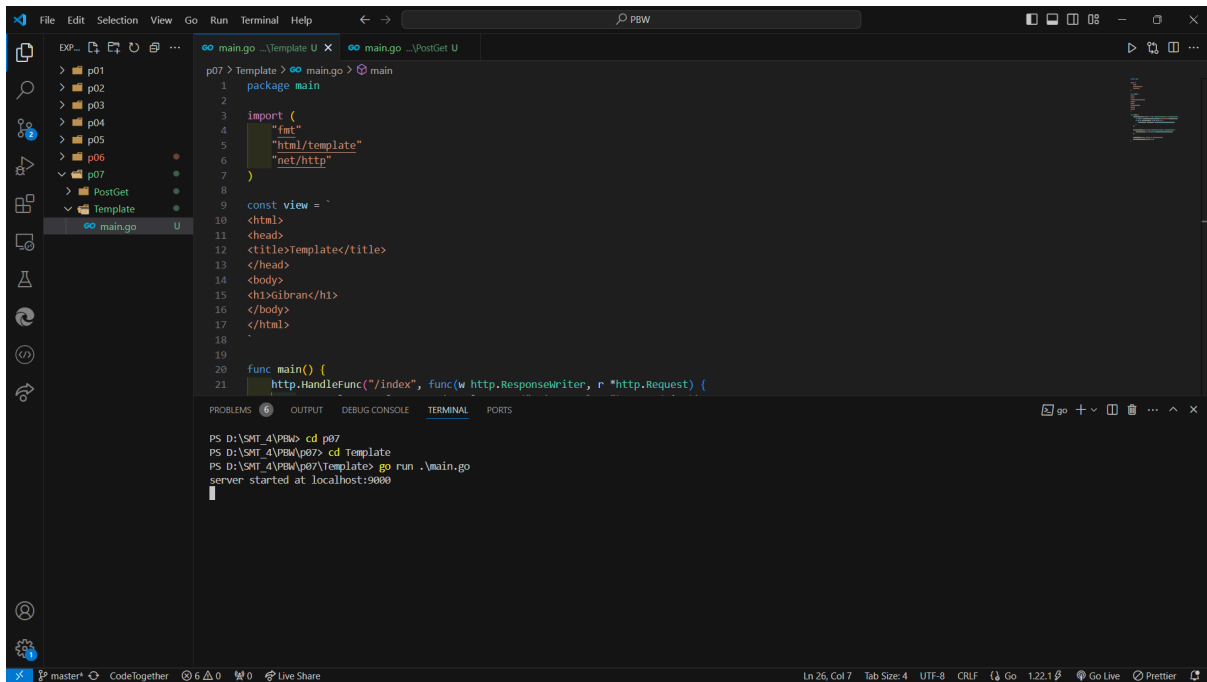
- Praktek
  - Siapkan file main.go dengan program sebagai berikut:



```
1 package main
2
3 import (
4     "fmt"
5     "html/template"
6     "net/http"
7 )
8
9 const view = `
10 <html>
11 <head>
12 <title>Template</title>
13 </head>
14 <body>
15 <h1>Gibran</h1>
16 </body>
17 </html>
18 `
19
20 func main() {
21     http.HandleFunc("/index", func(w http.ResponseWriter, r *http.Request) {
22         var tmpl = template.Must(template.New("main-template").Parse(view))
23         if err := tmpl.Execute(w, nil); err != nil {
24             http.Error(w, err.Error(), http.StatusInternalServerError)
25         }
26     })
27
28     http.HandleFunc("/", func(w http.ResponseWriter, r *http.Request) {
29         http.Redirect(w, r, "/index", http.StatusTemporaryRedirect)
30     })
31
32     fmt.Println("server started at localhost:9000")
33     http.ListenAndServe(":9000", nil)
34 }
35
```

- Testing

- Lakukan testing dengan run program menggunakan perintah “go run .\main.go”



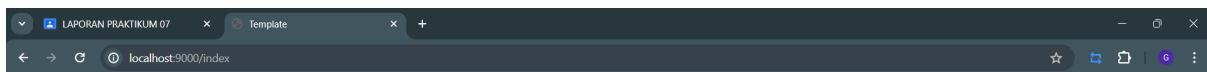
The screenshot shows the Visual Studio Code interface. The editor displays a Go file named `main.go` with the following code:

```
1 package main
2
3 import (
4     "fmt"
5     "html/template"
6     "net/http"
7 )
8
9 const view = `
10 <html>
11 <head>
12 <title>Template</title>
13 </head>
14 <body>
15 <h1>Gibran</h1>
16 </body>
17 </html>
18 `
19
20 func main() {
21     http.HandleFunc("/index", func(w http.ResponseWriter, r *http.Request) {
```

The terminal at the bottom shows the following commands and output:

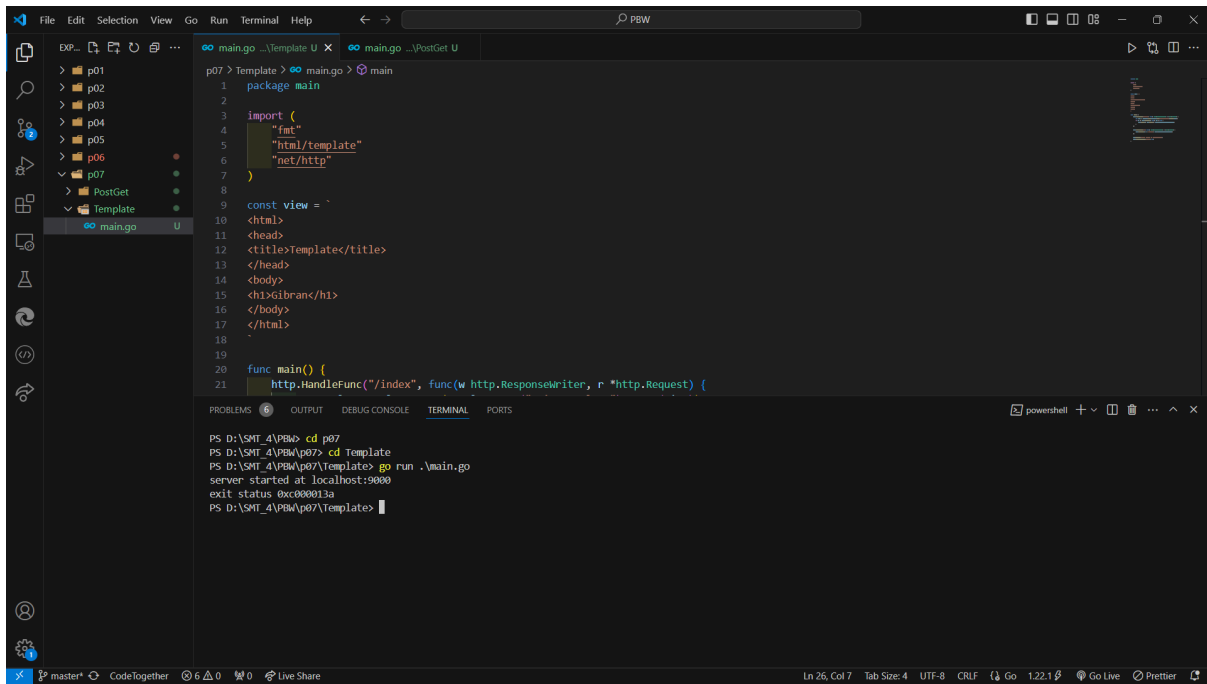
```
PS D:\SMT_A\PBW> cd p07
PS D:\SMT_A\PBW\p07> cd Template
PS D:\SMT_A\PBW\p07\template> go run .\main.go
server started at localhost:9000
```

- Copy localhost di terminal dan letakan pada url kosong untuk melihat outputnya



Hello

- Jika ingin merubah isi file, lakukan outConnections dengan server terlebih dahulu dengan menekan CTRL + C pada terminal.



```
1 package main
2
3 import (
4     "fmt"
5     "html/template"
6     "net/http"
7 )
8
9 const view = `
10 <html>
11 <head>
12 <title>Template</title>
13 </head>
14 <body>
15 <h1>Gibran</h1>
16 </body>
17 </html>
18 `
19
20 func main() {
21     http.HandleFunc("/index", func(w http.ResponseWriter, r *http.Request) {
```

```
PS D:\SMT_4\PBW> cd p07
PS D:\SMT_4\PBW\p07> cd Template
PS D:\SMT_4\PBW\p07\Template> go run .\main.go
server started at localhost:9000
exit status 0xc00013a
PS D:\SMT_4\PBW\p07\Template>
```

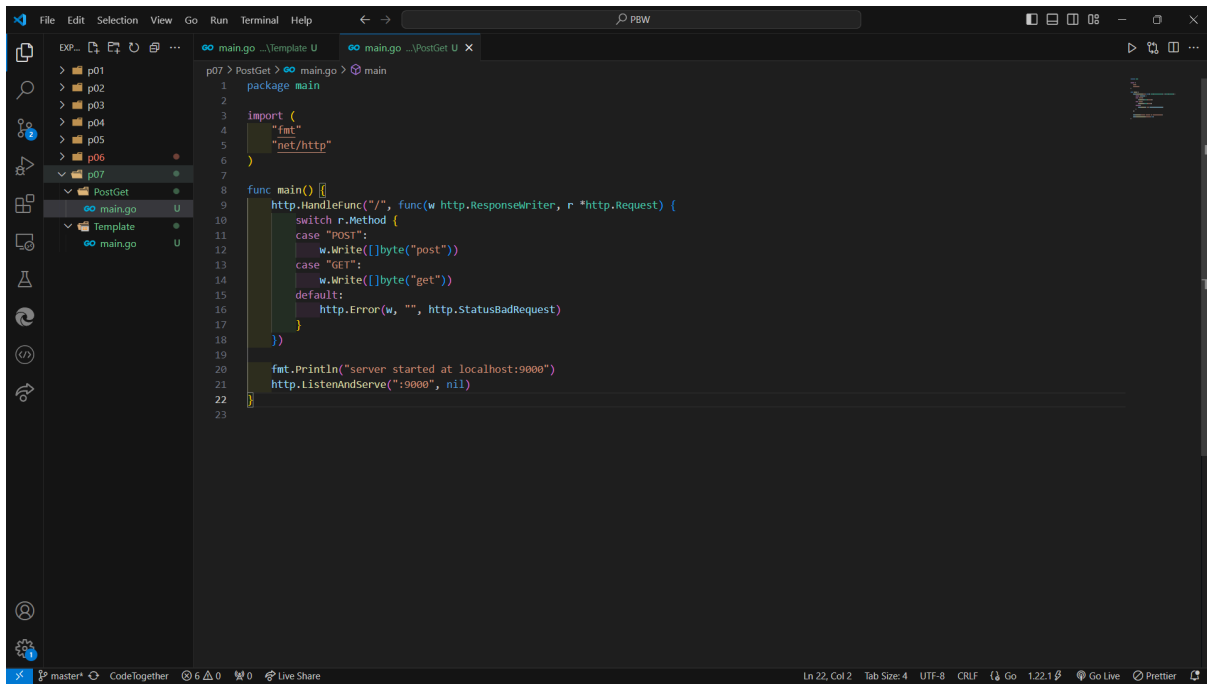
## 2. HTTP METHOD: POST & GET

- Penjelasan

Pengembang perangkat lunak dapat menggunakan Postman sebagai alat utama untuk mengelola dan menguji APIs. Dengan platform ini, pengguna dapat mengirim permintaan HTTP ke server, menangani respons, dan menguji berbagai metode permintaan, seperti GET dan POST. Contoh kode diatas menunjukkan implementasi sederhana untuk menangani permintaan GET dan POST di server menggunakan bahasa pemrograman Go. Selain itu, Postman memungkinkan pengembang untuk menguji endpoint dengan menggunakannya. Postman menjadi alat yang tak tergantikan dalam proses pengembangan perangkat lunak modern berkat antarmuka yang mudah digunakan.

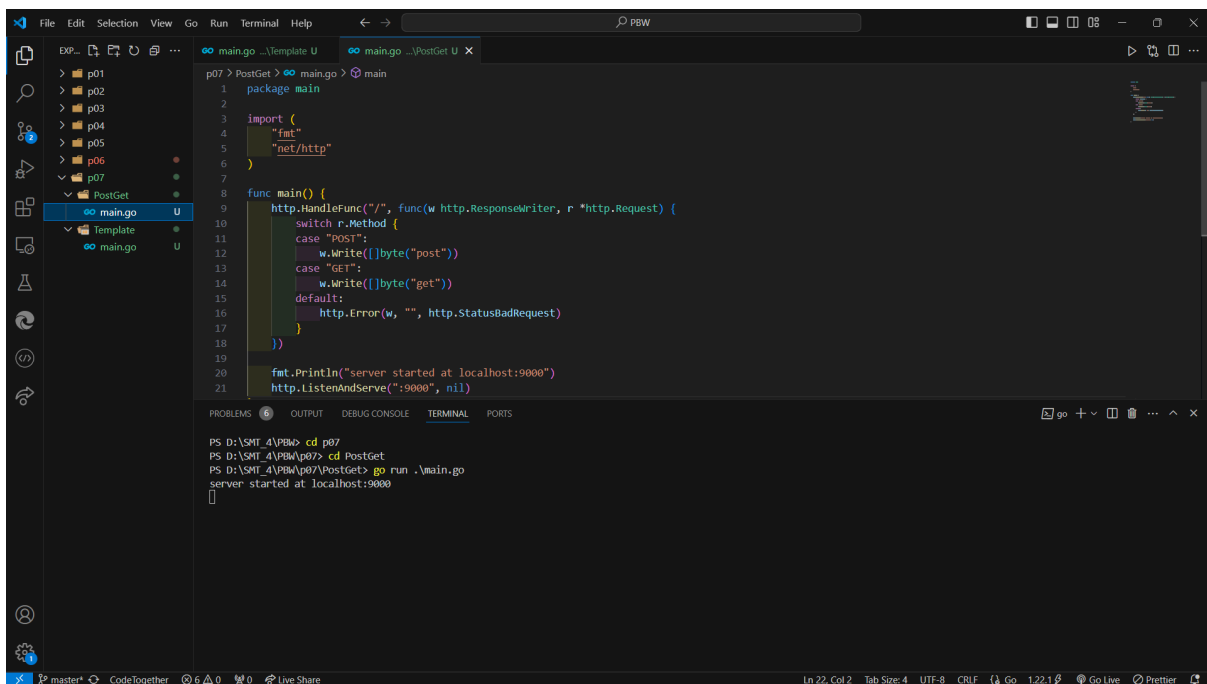
- Praktek

Buatlah file dengan nama main.go yang berisi code sebagai berikut:



- Testing

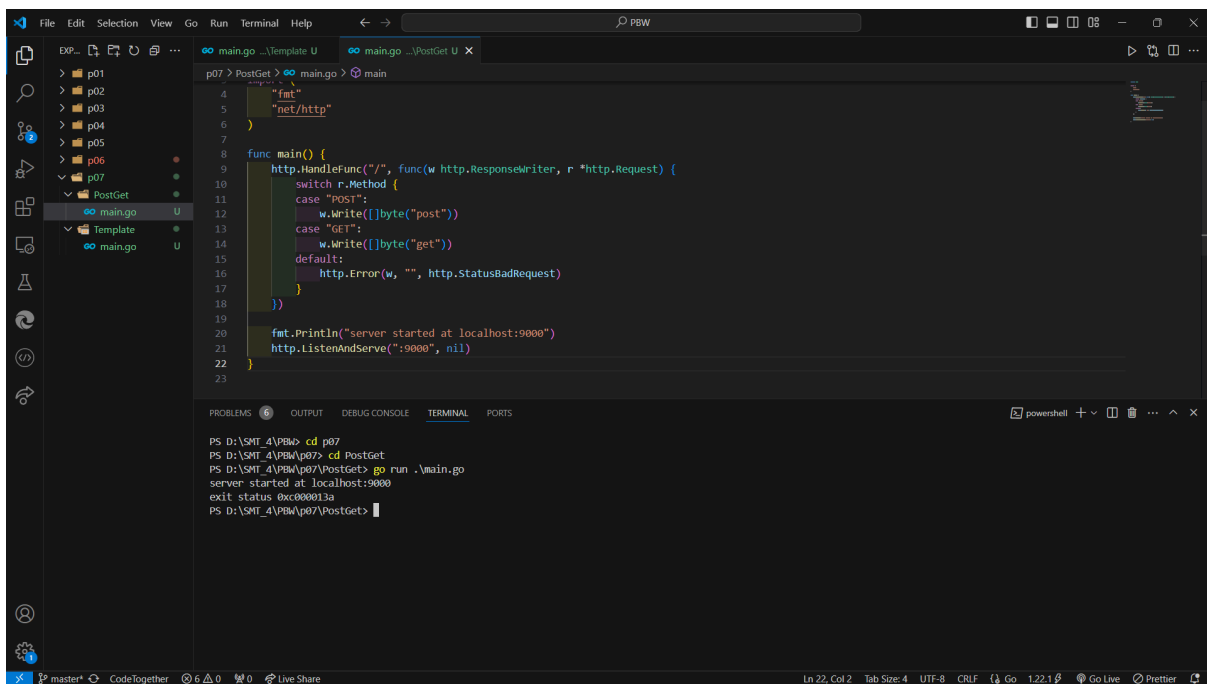
- Lakukan testing dengan run program menggunakan perintah “go run .\(namafile).go”



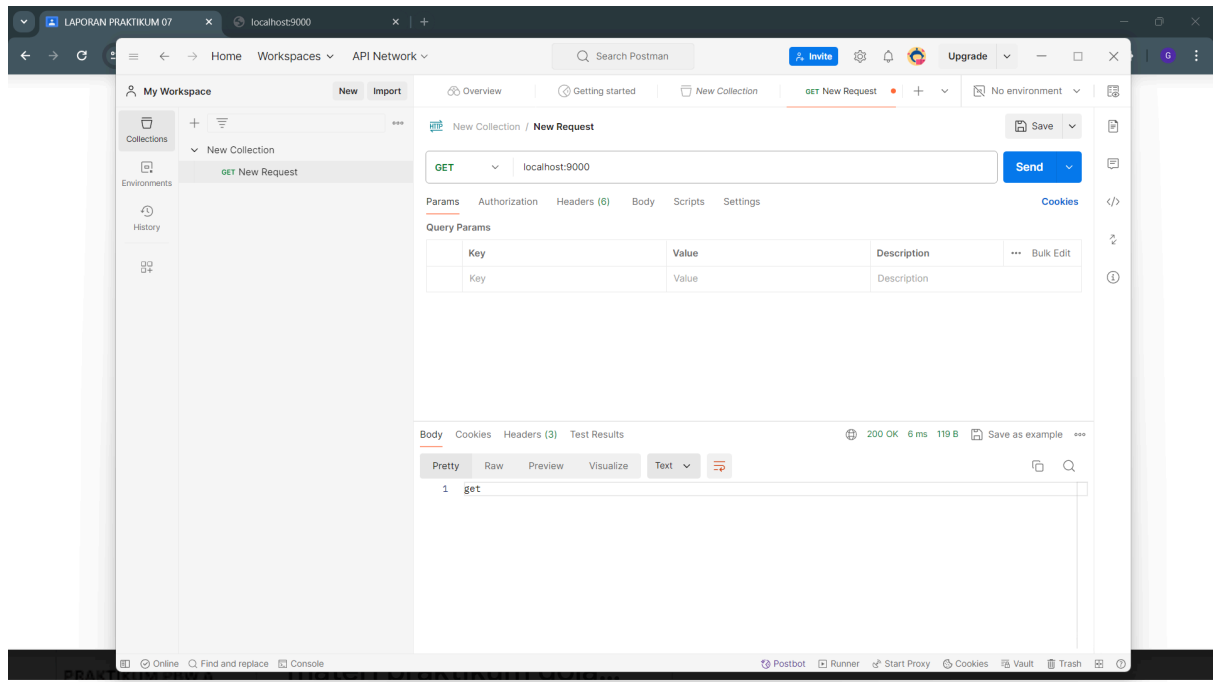
- Copy localhost di terminal dan letakan pada url kosong untuk melihat outputnya



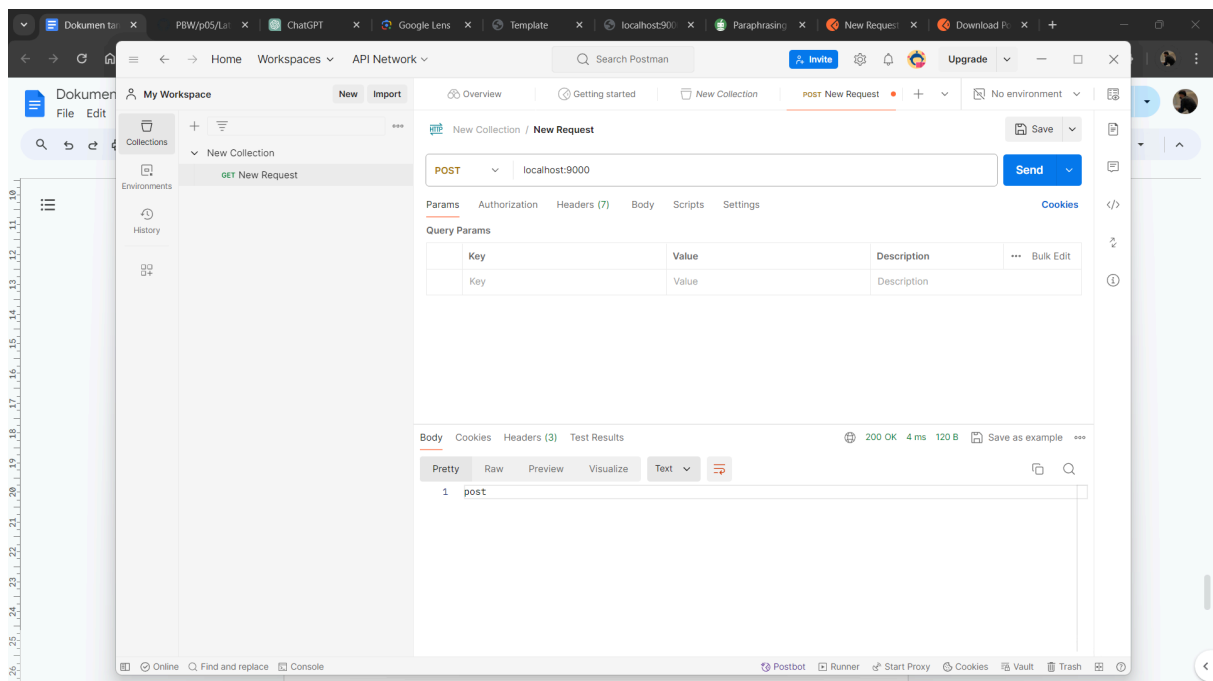
- Jika ingin merubah isi file, lakukan outConnections dengan server terlebih dahulu dengan menekan CTRL + C pada terminal.



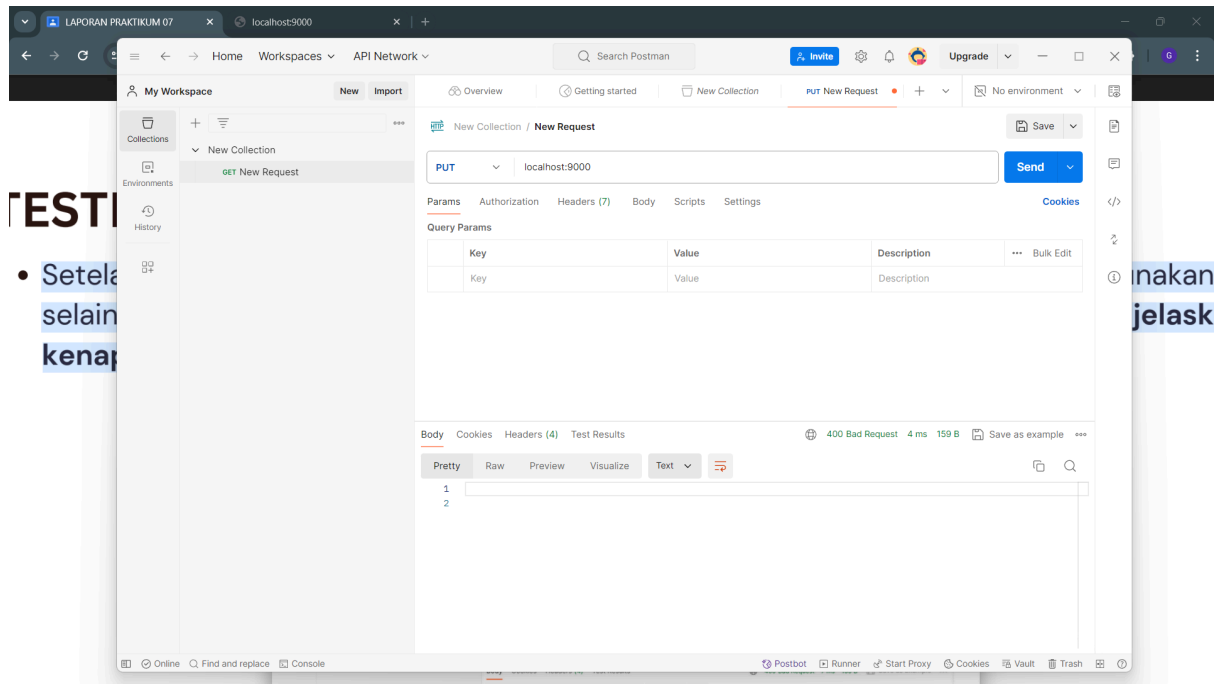
- Lakukan testing menggunakan method GET jika berhasil akan menunjukan 'get'



- Lakukan testing menggunakan method POST, jika berhasil akan menunjukan 'post'



- Karena server yang digunakan di dalam program hanya mendukung dua metode HTTP, yaitu GET dan POST. Pesan "400 Bad Request" muncul ketika server menerima permintaan yang tidak dapat diproses karena formatnya tidak valid atau tidak dikenali. Hal terjadi ketika permintaan yang dikirim tidak sesuai dengan aturan atau spesifikasi yang telah ditetapkan oleh server.



### 3. FORM VALUE

- Penjelasan

Front end adalah bagian dari sebuah aplikasi atau situs web yang dapat dilihat dan diakses langsung oleh pengguna. Ini mencakup tampilan dan interaksi langsung dengan pengguna menggunakan teknologi seperti HTML, CSS, dan JavaScript. HTML menentukan struktur dasar halaman web, sedangkan CSS mengatur tampilan dan gaya halaman, dan JavaScript menambahkan interaktivitas seperti animasi dan validasi formulir. Dengan kombinasi teknologi ini, front end menghasilkan antarmuka antarmuka.

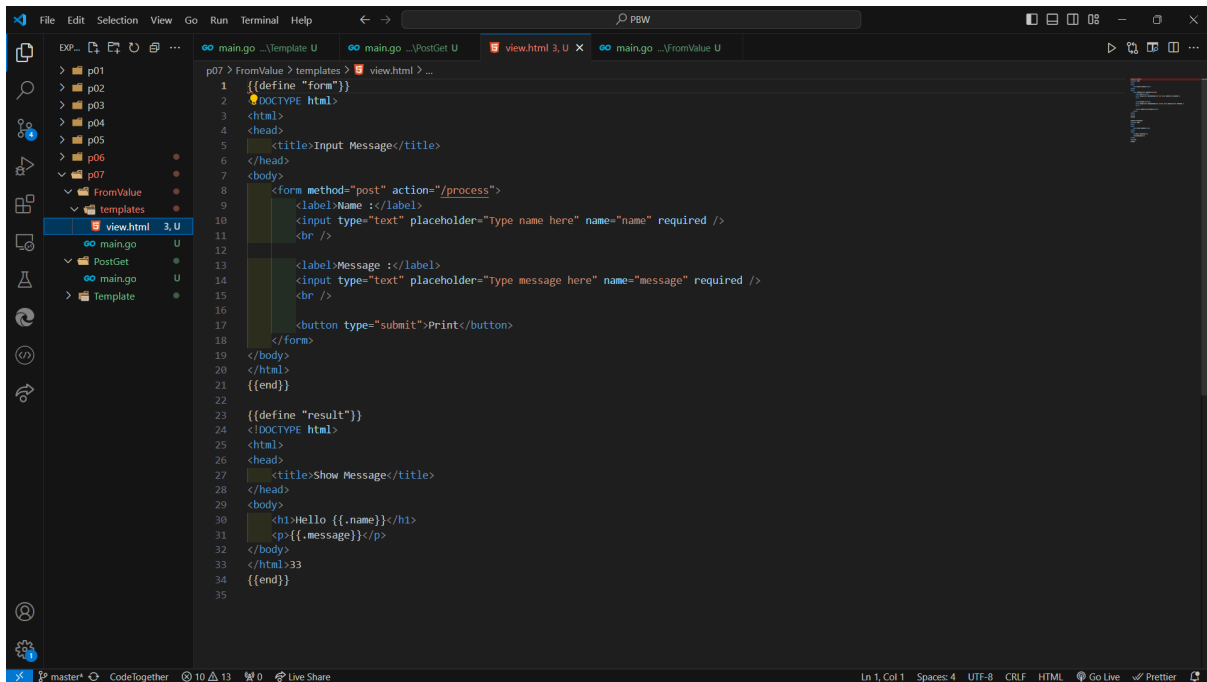
Back end adalah bagian penting dari sebuah aplikasi atau situs web yang berjalan di sisi server. Ini melibatkan pengolahan data, manajemen database, autentikasi pengguna, dan implementasi logika bisnis menggunakan berbagai bahasa pemrograman dan framework. Dengan back end yang solid, aplikasi dapat menyediakan layanan yang handal dan aman kepada pengguna serta menjaga integritas data dengan baik.

API (Application Programming Interface) adalah protokol dan aturan yang memungkinkan aplikasi berinteraksi satu sama lain dan berbagi informasi. API memungkinkan aplikasi terintegrasi satu sama lain dan mendapatkan akses ke data dan fitur dari platform tertentu, dan memungkinkan aplikasi berintegrasi dengan aplikasi lain.



- Praktek (FRONT END)

- Buatlah folder templates dan buat file view.html dengan isi sebagai berikut:

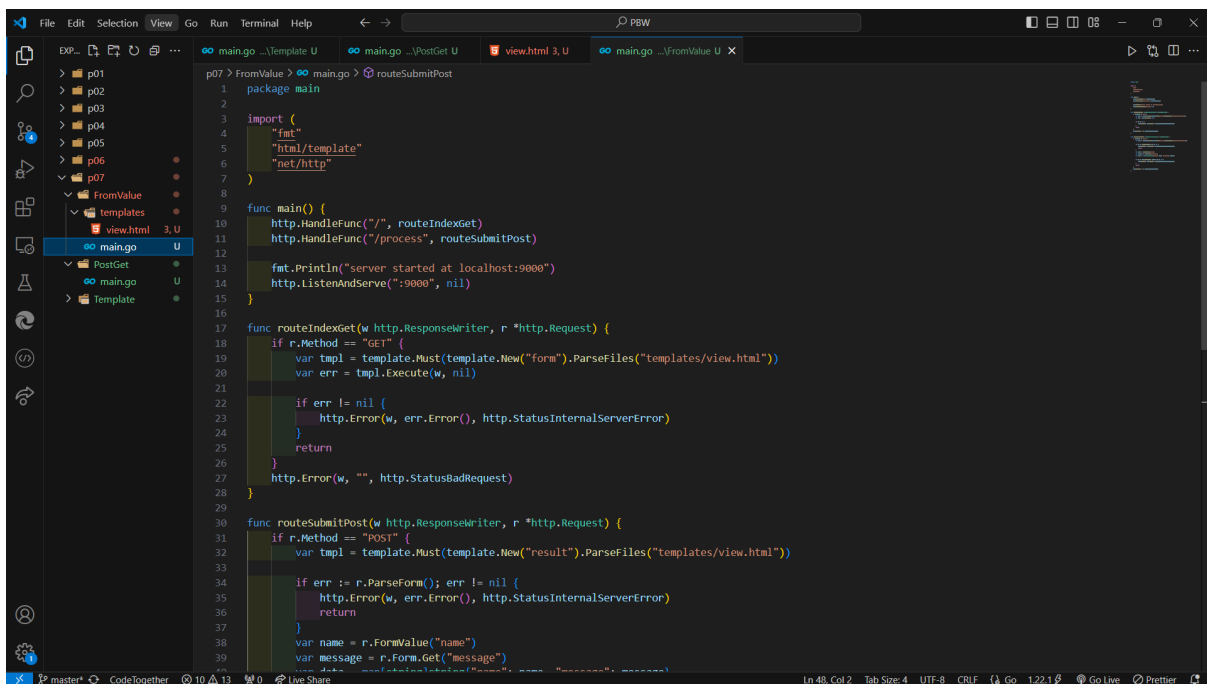


```

1  {{define "form"}}
2  <doctype html>
3  <html>
4  <head>
5      <title>Input Message</title>
6  </head>
7  <body>
8      <form method="post" action="/process">
9          <label>Name :</label>
10         <input type="text" placeholder="Type name here" name="name" required />
11         <br />
12         <label>Message :</label>
13         <input type="text" placeholder="Type message here" name="message" required />
14         <br />
15         <button type="submit">Print</button>
16     </form>
17 </body>
18 </html>
19 {{end}}
20
21 {{define "result"}}
22 <doctype html>
23 <html>
24 <head>
25     <title>Show Message</title>
26 </head>
27 <body>
28     <h1>Hello {{.name}}</h1>
29     <p>{{.message}}</p>
30 </body>
31 </html>
32 {{end}}
33
34

```

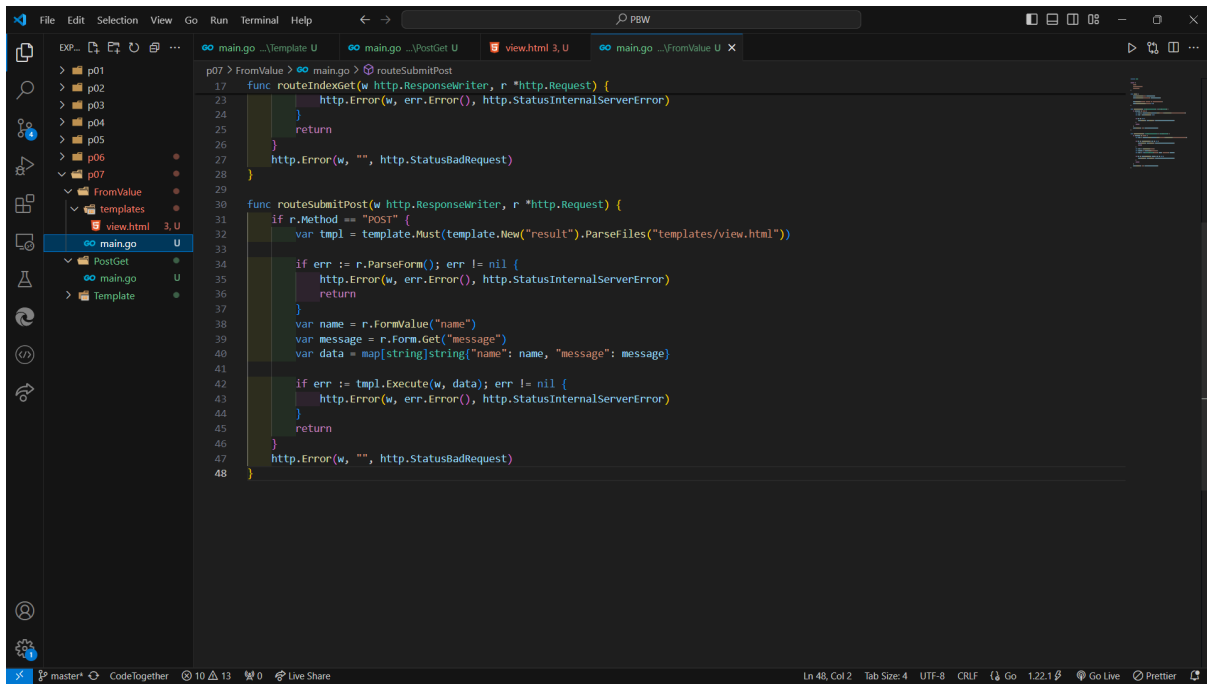
- Buat file main.go di dalam folder FromValue dengan isi code berikut



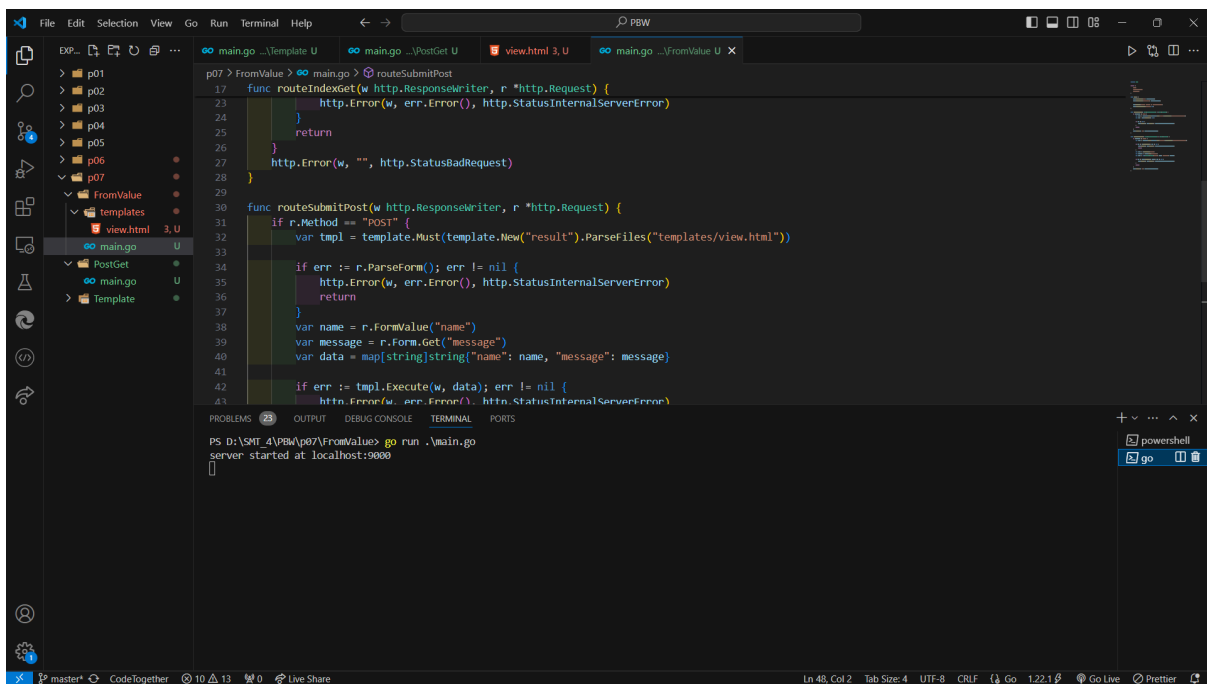
```

1  package main
2
3  import (
4      "fmt"
5      "html/template"
6      "net/http"
7  )
8
9  func main() {
10     http.HandleFunc("/", routeIndexGet)
11     http.HandleFunc("/process", routeSubmitPost)
12
13     fmt.Println("server started at localhost:9000")
14     http.ListenAndServe(":9000", nil)
15 }
16
17 func routeIndexGet(w http.ResponseWriter, r *http.Request) {
18     if r.Method == "GET" {
19         var tmpl = template.Must(template.New("form").ParseFiles("templates/view.html"))
20         var err = tmpl.Execute(w, nil)
21
22         if err != nil {
23             http.Error(w, err.Error(), http.StatusInternalServerError)
24         }
25         return
26     }
27     http.Error(w, "", http.StatusBadRequest)
28 }
29
30 func routeSubmitPost(w http.ResponseWriter, r *http.Request) {
31     if r.Method == "POST" {
32         var tmpl = template.Must(template.New("result").ParseFiles("templates/view.html"))
33
34         if err := r.ParseForm(); err != nil {
35             http.Error(w, err.Error(), http.StatusInternalServerError)
36             return
37         }
38         var name = r.FormValue("name")
39         var message = r.FormValue("message")
40         // fmt.Println(name, message)

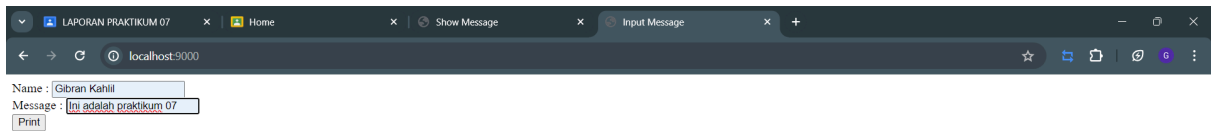
```



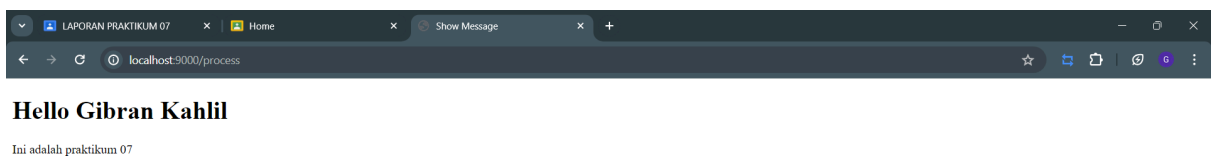
- Testing
  - Running file `main.go` di dalam terminal



- buka localhost di terminal di dalam web dan isi Nama & Message



- 
- Hasil Pengissian setelah di print



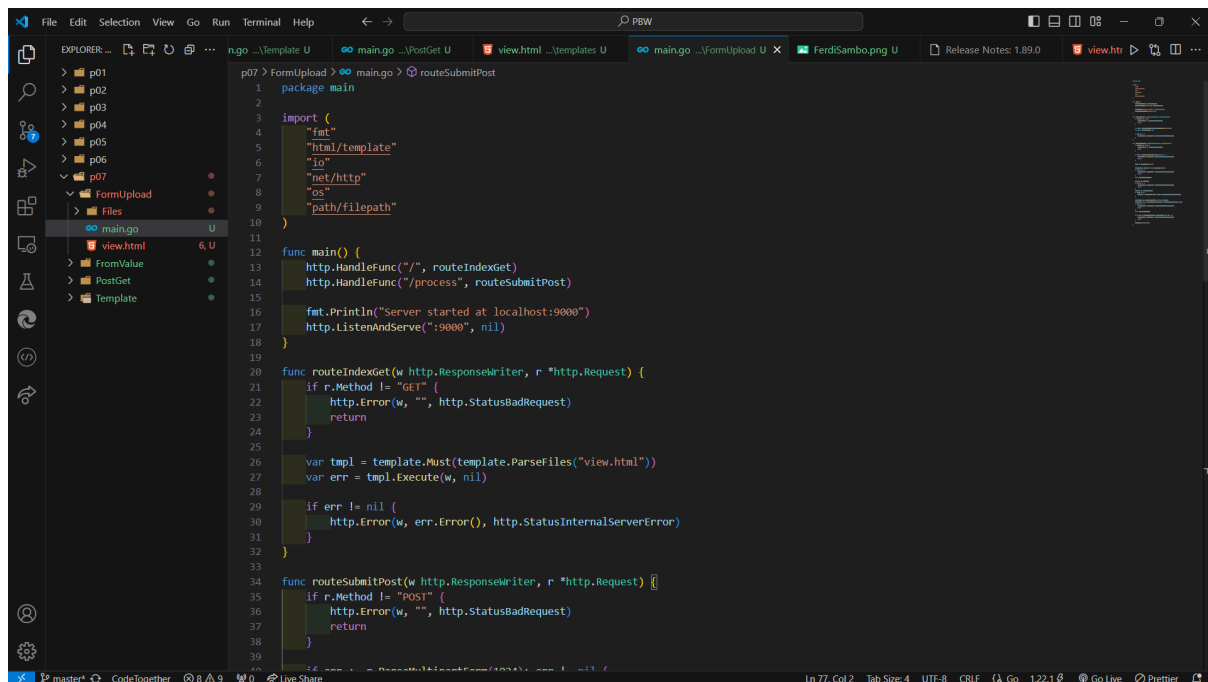
#### 4. FORM UPLOAD FILE

- Penjelasan

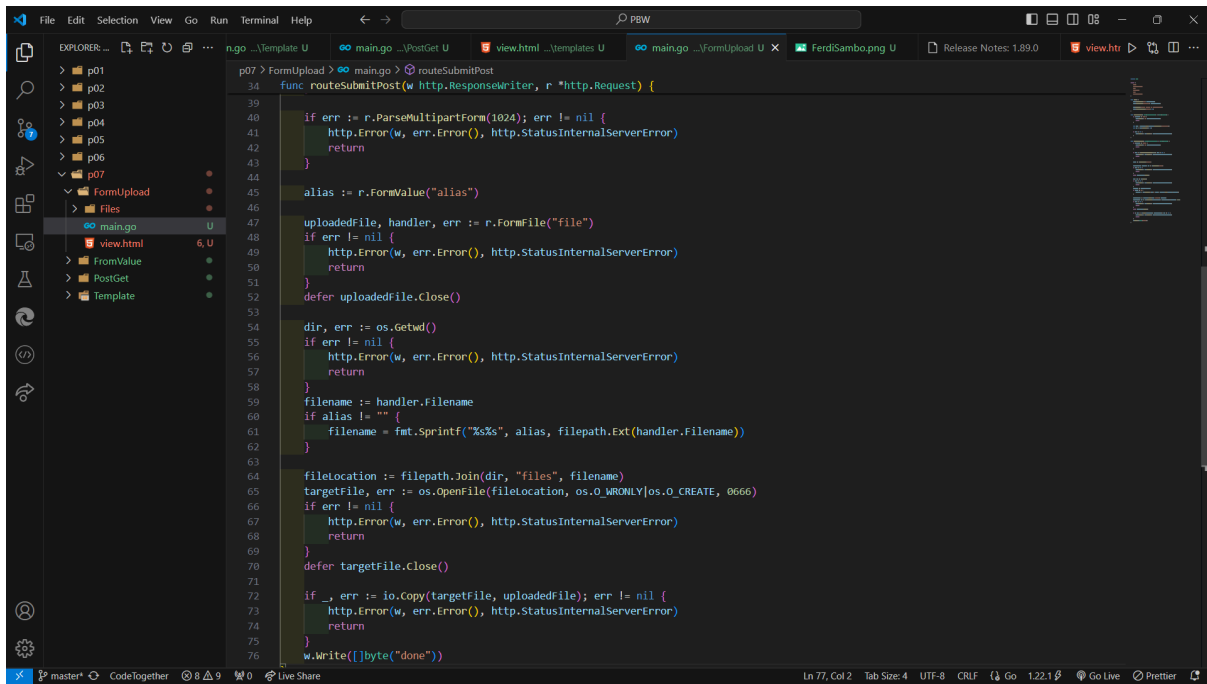
Ketika menggunakan metode POST untuk mengirimkan data formulir yang mencakup file ke server, atribut `enctype="multipart/form-data"` pada tag formulir sangat penting. Tanpa atribut ini, data file tidak akan terkirim dengan benar, dan server tidak akan

dapat mengidentifikasi bahwa ada file yang dikirim bersamaan dengan formulir. Ini adalah hasil dari cara browser mengirimkan formulir ke server. Ketika atribut enctype tidak ditentukan, browser secara default menggunakan enctype dengan nilai application/x-www-form-urlencoded. Ini mengkodekan data formulir menjadi format yang dapat dibaca server. Karena format ini tidak memungkinkan pengiriman data biner, itu tidak cocok untuk mengirim file. Dengan menambahkan enctype="multipart/form-data", browser menyadari bahwa formulir ini akan mengandung file yang akan dikirimkan bersamaan dengan teks biasa. Ini memungkinkan browser untuk menggunakan. Ini memungkinkan browser untuk menggunakan mekanisme pengiriman data yang sesuai dengan mengirimkan data biner yang diperlukan untuk file tersebut. Jadi, penting untuk selalu menyertakan enctype="multipart/form-data" pada tag <form> ketika Anda memiliki input file dalam formulir HTML dan ingin mengirimkan file tersebut ke server melalui metode POST. Tanpa itu, file tidak akan terkirim dengan benar, dan server mungkin tidak dapat memproses data yang diterima dengan benar.

- Praktek (FRONT END)
  - Buatlah folder files serta buat file main.go dengan isi sebagai berikut:

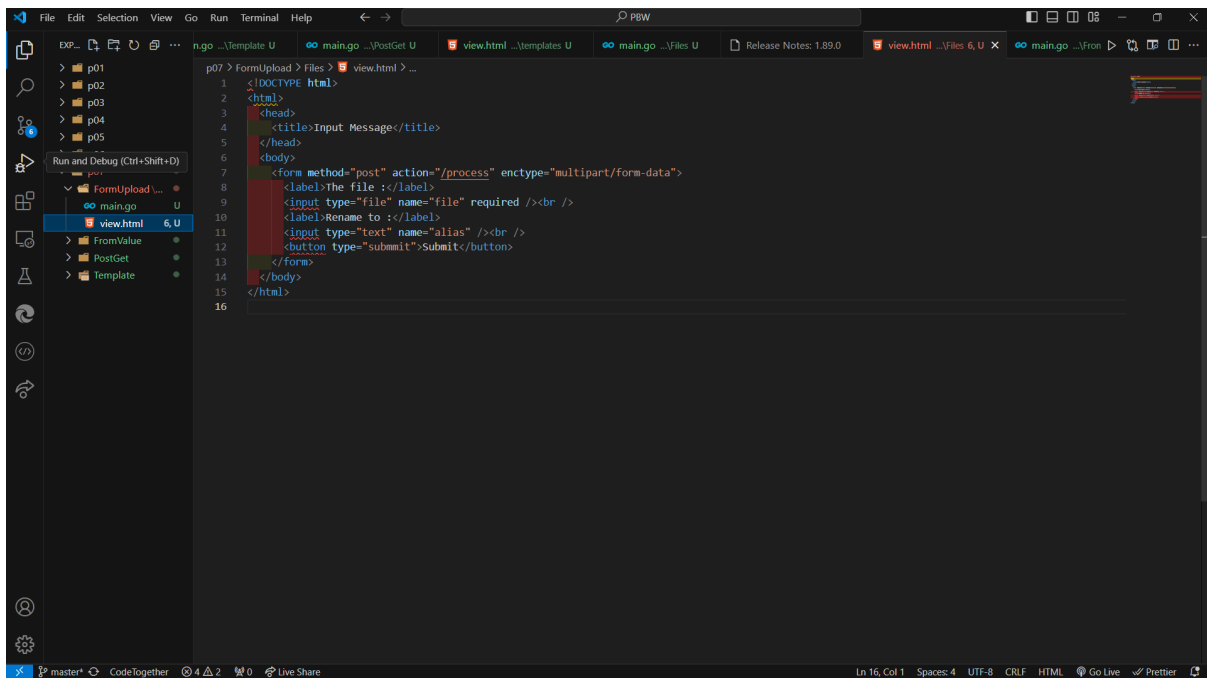


```
1 package main
2
3 import (
4     "fmt"
5     "html/template"
6     "io"
7     "net/http"
8     "os"
9     "path/filepath"
10 )
11
12 func main() {
13     http.HandleFunc("/", routeIndexGet)
14     http.HandleFunc("/process", routeSubmitPost)
15
16     fmt.Println("Server started at localhost:9000")
17     http.ListenAndServe(":9000", nil)
18 }
19
20 func routeIndexGet(w http.ResponseWriter, r *http.Request) {
21     if r.Method != "GET" {
22         http.Error(w, "", http.StatusBadRequest)
23         return
24     }
25
26     var tmpl = template.Must(template.ParseFiles("view.html"))
27     var err = tmpl.Execute(w, nil)
28
29     if err != nil {
30         http.Error(w, err.Error(), http.StatusInternalServerError)
31     }
32 }
33
34 func routeSubmitPost(w http.ResponseWriter, r *http.Request) {}
```



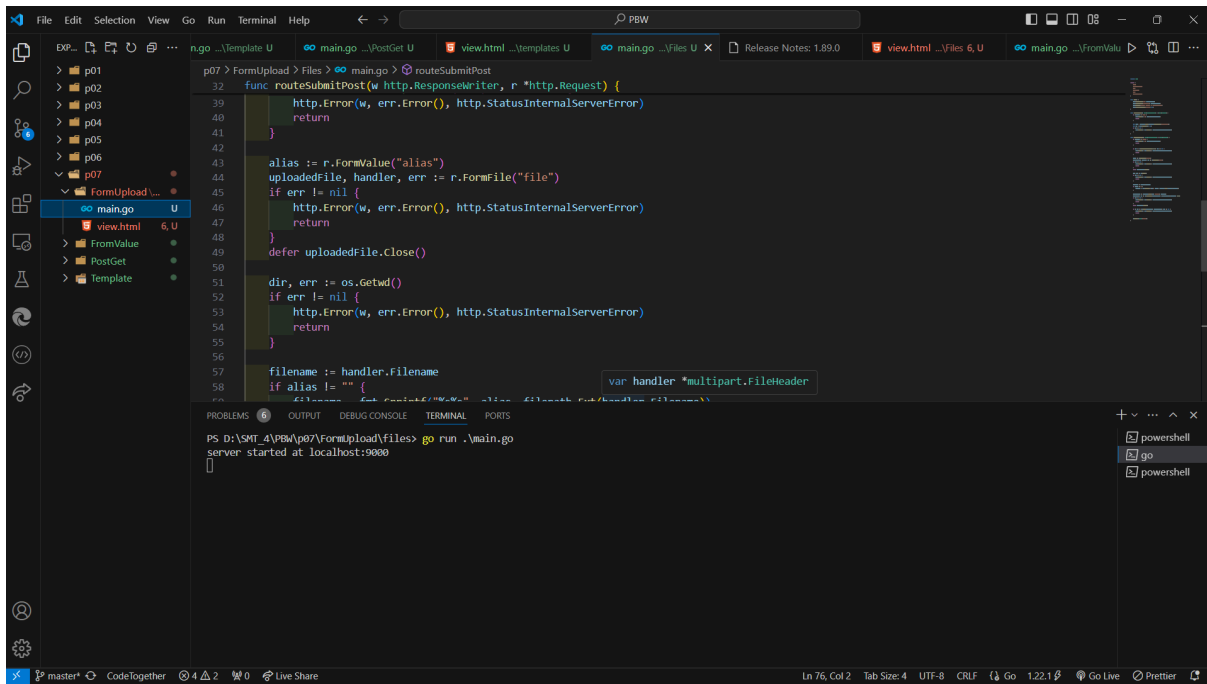
```
34 func routeSubmitPost(w http.ResponseWriter, r *http.Request) {
35
36     if err := r.ParseMultipartForm(1024); err != nil {
37         http.Error(w, err.Error(), http.StatusInternalServerError)
38         return
39     }
40
41     alias := r.FormValue("alias")
42
43     uploadedFile, handler, err := r.FormFile("file")
44     if err != nil {
45         http.Error(w, err.Error(), http.StatusInternalServerError)
46         return
47     }
48     defer uploadedFile.Close()
49
50     dir, err := os.Getwd()
51     if err != nil {
52         http.Error(w, err.Error(), http.StatusInternalServerError)
53         return
54     }
55     filename := handler.Filename
56     if alias != "" {
57         filename = fmt.Sprintf("%s%s", alias, filepath.Ext(handler.Filename))
58     }
59
60     fileLocation := filepath.Join(dir, "files", filename)
61     targetFile, err := os.OpenFile(fileLocation, os.O_WRONLY|os.O_CREATE, 0666)
62     if err != nil {
63         http.Error(w, err.Error(), http.StatusInternalServerError)
64         return
65     }
66     defer targetFile.Close()
67
68     if _, err := io.Copy(targetFile, uploadedFile); err != nil {
69         http.Error(w, err.Error(), http.StatusInternalServerError)
70         return
71     }
72     w.Write([]byte("done"))
73 }
```

- Buatlah folder files serta buat file view.html dengan isi sebagai berikut:

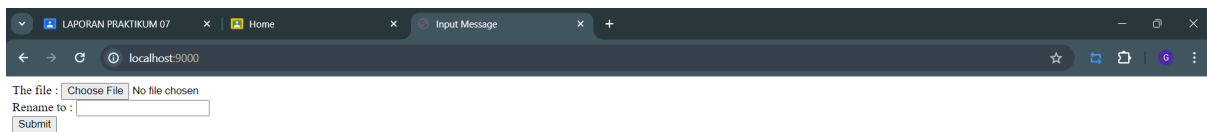


```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Input Message</title>
5 </head>
6 <body>
7 <form method="post" action="/process" enctype="multipart/form-data">
8 <label>The file :</label>
9 <input type="file" name="file" required /><br />
10 <label>Rename to :</label>
11 <input type="text" name="alias" /><br />
12 <button type="submit">Submit</button>
13 </form>
14 </body>
15 </html>
16
```

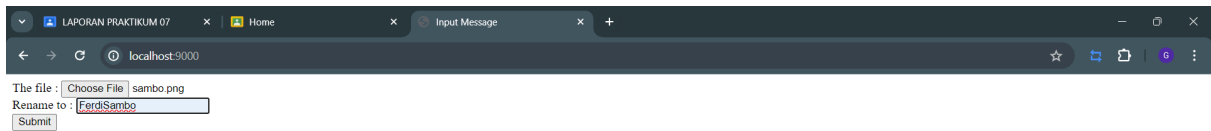
- Testing
  - Running file main.go di dalam terminal



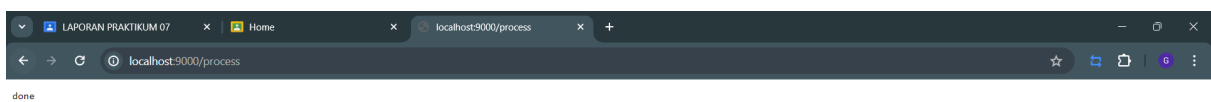
- buka localhost di terminal di dalam web



- Isi file dan nama file yang ingin diubah



- Hasil Pengisian setelah di submit



Kesimpulan: Dari penjelasan di atas, jelas bahwa parsing HTML adalah proses penting dalam pengembangan web. Front end dan back end adalah dua komponen utama dalam pengembangan aplikasi web, dengan API berfungsi untuk menghubungkan berbagai aplikasi atau layanan satu sama lain. Untuk mengirimkan data formulir yang mencakup file ke server

menggunakan metode POST, atribut "enctype="multipart/form-data" pada tag "form" sangat penting. Memahami konsep-konsep ini membantu mengembangkan aplikasi web yang berhasil.