

**LAPORAN LATIHAN PRAKTIKUM KE-8**  
**PEMROGRAMAN BERBASIS WEB**  
**KELAS A**

Disusun Untuk Memenuhi Tugas Mata Kuliah Prak. Pemrograman Berbasis Web



Disusun oleh:

Arsya Yan Duribta                            4522210117

Dosen Pengampu:

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

**Program Studi Teknik Informatika**  
**Fakultas Teknik Universitas Pancasila**  
**2023/2024**

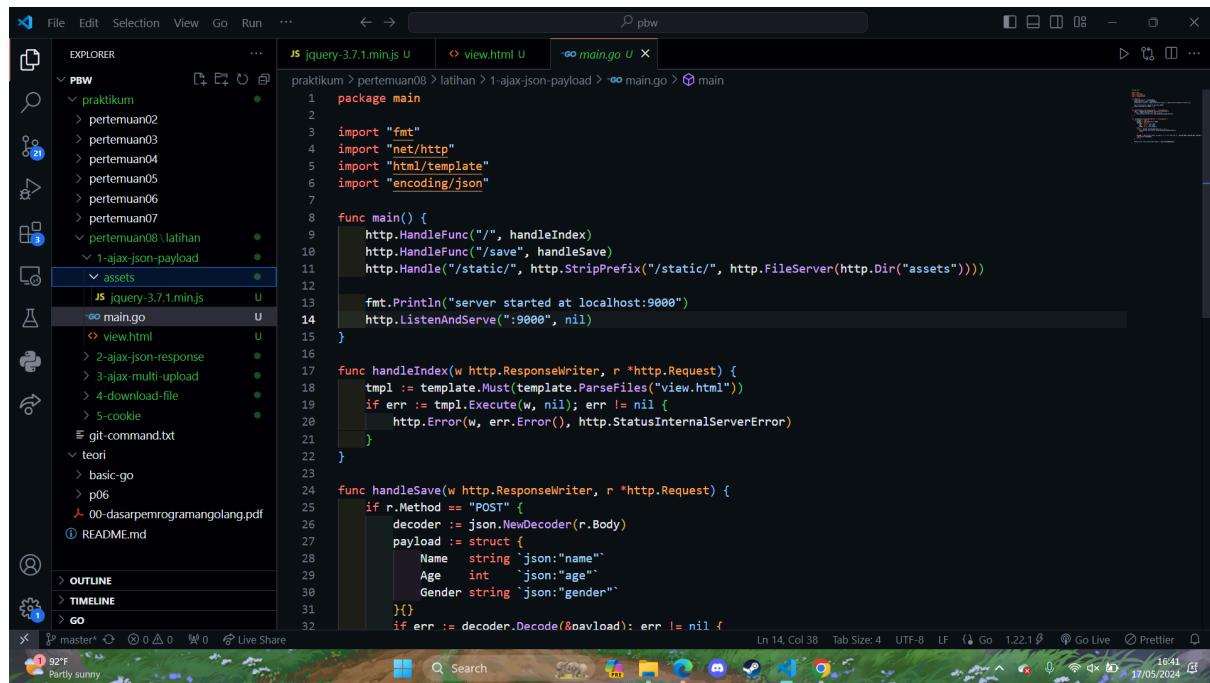
## Link Repository Github:

<https://github.com/Arsyayd11/pbw/tree/master/praktikum/pertemuan08/latihan>

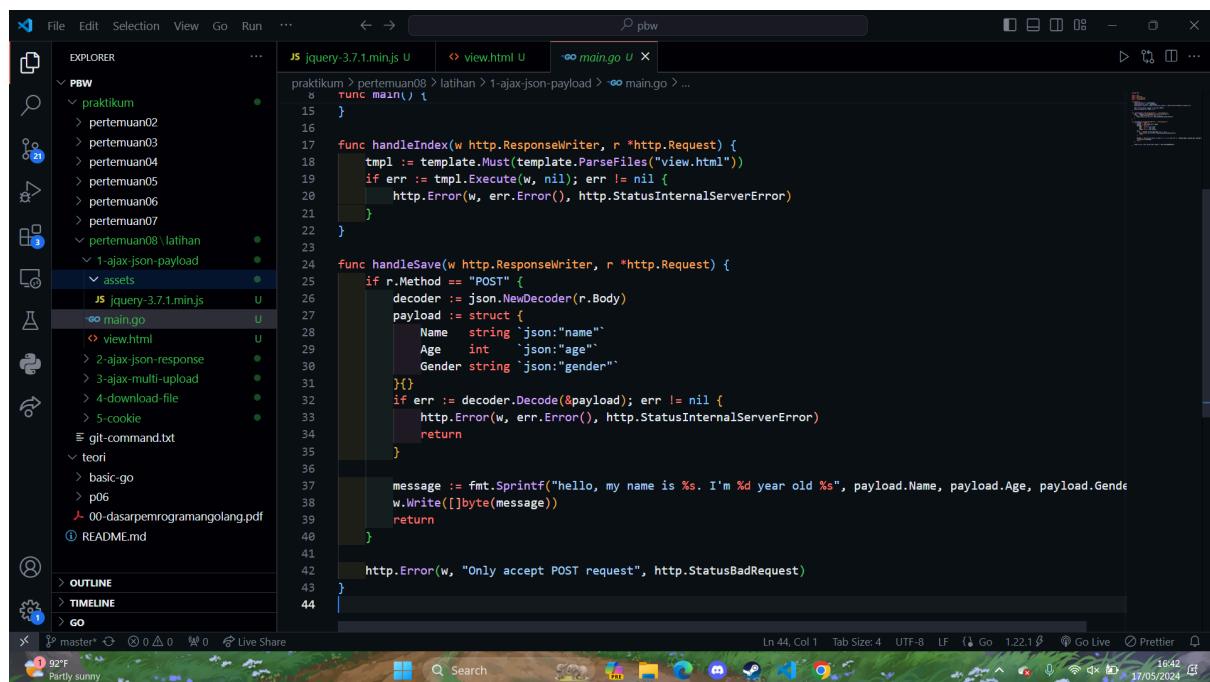
### Latihan 1: 1-ajax-json-payload

Implementasinya (Source Code):

- main.go



```
File Edit Selection View Go Run ... ← → pbw
EXPLORER JS jquery-3.7.1.min.js U view.html U main.go ...
praktikum > pertemuan08 > latihan > 1-ajax-json-payload > main.go
1 package main
2
3 import "fmt"
4 import "net/http"
5 import "html/template"
6 import "encoding/json"
7
8 func main() {
9     http.HandleFunc("/", handleIndex)
10    http.HandleFunc("/save", handleSave)
11    http.Handle("/static/", http.StripPrefix("/static/", http.FileServer(http.Dir("assets"))))
12
13    fmt.Println("server started at localhost:9000")
14    http.ListenAndServe(":9000", nil)
15 }
16
17 func handleIndex(w http.ResponseWriter, r *http.Request) {
18     tmpl := template.Must(template.ParseFiles("view.html"))
19     if err := tmpl.Execute(w, nil); err != nil {
20         http.Error(w, err.Error(), http.StatusInternalServerError)
21     }
22 }
23
24 func handleSave(w http.ResponseWriter, r *http.Request) {
25     if r.Method == "POST" {
26         decoder := json.NewDecoder(r.Body)
27         payload := struct {
28             Name string `json:"name"`
29             Age int `json:"age"`
30             Gender string `json:"gender"`
31         }()
32         if err := decoder.Decode(&payload); err != nil {
33             http.Error(w, err.Error(), http.StatusInternalServerError)
34         }
35     }
36
37     message := fmt.Sprintf("hello, my name is %s. I'm %d year old %s", payload.Name, payload.Age, payload.Gender)
38     w.Write([]byte(message))
39     return
40 }
41
42 http.Error(w, "Only accept POST request", http.StatusBadRequest)
43
44 }
```



```
File Edit Selection View Go Run ... ← → pbw
EXPLORER JS jquery-3.7.1.min.js U view.html U main.go ...
praktikum > pertemuan08 > latihan > 1-ajax-json-payload > main.go ...
8 func main() {
15 }
16
17 func handleIndex(w http.ResponseWriter, r *http.Request) {
18     tmpl := template.Must(template.ParseFiles("view.html"))
19     if err := tmpl.Execute(w, nil); err != nil {
20         http.Error(w, err.Error(), http.StatusInternalServerError)
21     }
22 }
23
24 func handleSave(w http.ResponseWriter, r *http.Request) {
25     if r.Method == "POST" {
26         decoder := json.NewDecoder(r.Body)
27         payload := struct {
28             Name string `json:"name"`
29             Age int `json:"age"`
30             Gender string `json:"gender"`
31         }()
32         if err := decoder.Decode(&payload); err != nil {
33             http.Error(w, err.Error(), http.StatusInternalServerError)
34         }
35     }
36
37     message := fmt.Sprintf("hello, my name is %s. I'm %d year old %s", payload.Name, payload.Age, payload.Gender)
38     w.Write([]byte(message))
39     return
40 }
41
42 http.Error(w, "Only accept POST request", http.StatusBadRequest)
43
44 }
```

- view.html

The screenshot shows a code editor interface with two tabs open: `view.html` and `main.go`. The `view.html` tab contains the following HTML and JavaScript code:

```
<!DOCTYPE html>
<html>
  <head>
    <title>JSON Payload</title>
    <script src="static/jquery-3.7.1.min.js"></script>
    <script>
      $(function () {
        $("#user-form").on("submit", function (e) {
          e.preventDefault();

          var $self = $(this);
          var payload = JSON.stringify({
            name: $('[name="name"]').val(),
            age: parseInt($('input[name="age"]').val(), 10),
            gender: $('input[name="gender"]').val()
          });

          $.ajax({
            url: $self.attr("action"),
            type: $self.attr("method"),
            data: payload,
            contentType: 'application/json',
          }).then(function (res) {
            $(".message").text(res);
          }).catch(function (a) {
            alert("ERROR: " + a.responseText);
          });
        });
      });
    </script>
  </head>
  <body>
```

The screenshot shows a code editor interface with two tabs open: `view.html` and `main.go`. The `view.html` tab contains the following HTML code:

```
<html>
  <body>
    <p class="message"></p>
    <form id="user-form" method="post" action="/save">
      <table border="1">
        <tr>
          <td>
            <label>Name :</label>
          </td>
          <td>
            <input required type="text" name="name" placeholder="Type name here" />
          </td>
        </tr>
        <tr>
          <td>
            <label>Age :</label>
          </td>
          <td>
            <input required type="number" name="age" placeholder="Set age" />
          </td>
        </tr>
        <tr>
          <td>
            <label>Gender :</label>
          </td>
          <td>
            <select name="gender" required style="width: 100%;">
              <option value="">Select one</option>
              <option value="male">Male</option>
              <option value="female">Female</option>
            </select>
          </td>
        </tr>
      </table>
    </form>
  </body>
</html>
```

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left lists a project structure under 'praktikum'. The 'view.html' file is open in the main editor area, displaying an HTML form for user input. The code includes fields for name, age (with placeholder 'Set age'), gender (with options Male and Female), and a save button. The status bar at the bottom shows the file is 1643 bytes large and was last modified on 17/05/2024.

```
<html>
    <body>
        <form id="user-form" method="post" action="/save">
            <table border="1">
                <tr>
                    <td><input required type="text" name="name" placeholder="Name" /></td>
                    <td><input required type="number" name="age" placeholder="Set age" /></td>
                </tr>
                <tr>
                    <td><label>Gender :</label></td>
                    <td><select name="gender" required style="width: 100%;">
                        <option value="">Select one</option>
                        <option value="male">Male</option>
                        <option value="female">Female</option>
                    </select></td>
                </tr>
                <tr>
                    <td colspan="2" style="text-align: right;"><button type="submit">Save</button></td>
                </tr>
            </table>
        </form>
    </body>
</html>
```

Run code menggunakan terminal:

The screenshot shows the Visual Studio Code interface with the terminal tab active. The terminal window displays the output of a Go application named 'main.go'. The application starts a server at localhost:9999. The status bar at the bottom shows the file is 1641 bytes large and was last modified on 17/05/2024.

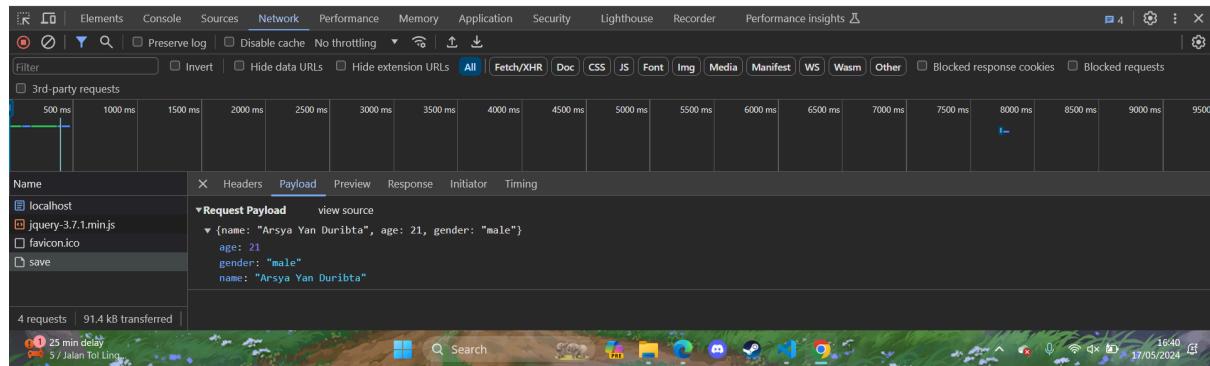
```
Arsyaa@LAPTOP-FKPF8QJI MINGW64 ~/Documents/pbw/praktikum/pertemuan08/latihan/1-ajax-json-payload (master)
$ go run main.go
server started at localhost:9999
```

Tampilan localhost pada website:



hello, my name is Arsy Yan Duribta. I'm 21 year old male

Name :   
Age :   
Gender :



## Latihan 2: 2-ajax-json-response

Implementasinya (Source Code):

A screenshot of a code editor (VS Code) showing a Go file named "main.go". The code implements an HTTP endpoint to handle JSON payloads. It defines a struct "Name" with fields "string" and "Age" (int). A slice of "Name" is populated with some values. The "ActionIndex" function uses "json.Marshal" to convert the struct to bytes and sets the "Content-Type" header to "application/json" before writing the response. The "main" function starts the server at "localhost:9000".

```
package main

import "fmt"
import "net/http"
import "encoding/json"

func ActionIndex(w http.ResponseWriter, r *http.Request) {
    data := []struct {
        Name string
        Age int
    }{
        {"Richard Grayson", 24},
        {"Jason Todd", 23},
        {"Tim Drake", 22},
        {"Damian Wayne", 21},
    }

    jsonInBytes, err := json.Marshal(data)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    w.Header().Set("Content-Type", "application/json")
    w.Write(jsonInBytes)
}

func main() {
    http.HandleFunc("/", ActionIndex)
    fmt.Println("server started at localhost:9000")
    http.ListenAndServe(":9000", nil)
}
```

The screenshot shows the Visual Studio Code interface with the following details:

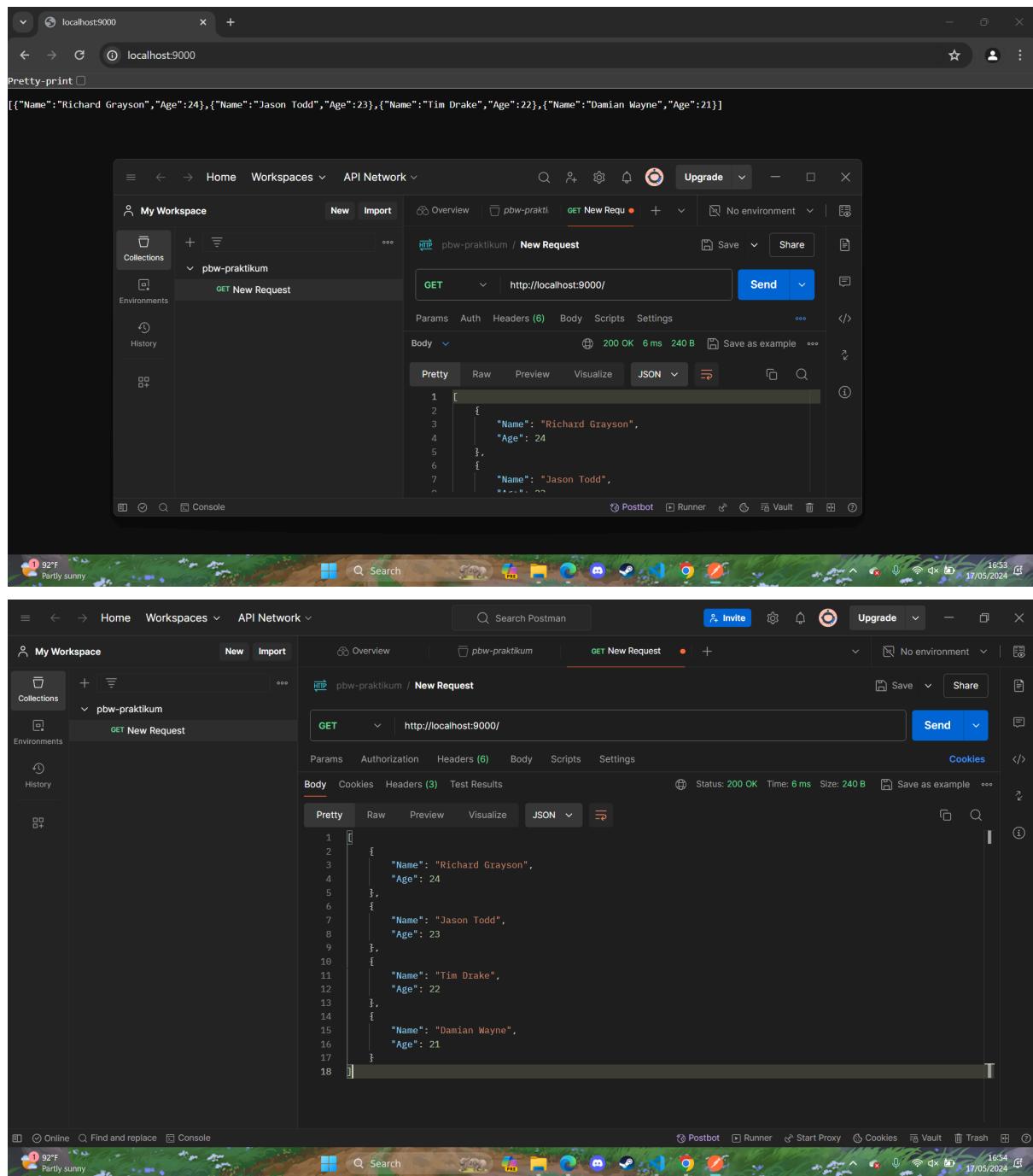
- File Explorer (Left):** Shows a project structure under "praktikum". The "main.go" file is selected.
- Code Editor (Center):** Displays the "main.go" code in Go language. The code defines a struct "data" with fields "Name" (string) and "Age" (int). It contains a slice of four people: Richard Grayson (24), Jason Todd (23), Tim Drake (22), and Damian Wayne (21). The code then marshals this data into JSON bytes and writes it to the response writer. The "main" function sets up a handler for the root path "/" to call the "ActionIndex" function.
- Terminal (Bottom):** Shows the command "go run main.go" being run in the terminal, resulting in the message "server started at localhost:9000".

Run code menggunakan terminal:

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows a project structure under "praktikum". The "main.go" file is selected.
- Code Editor (Center):** Displays the "main.go" code in Go language, identical to the previous screenshot.
- Terminal (Bottom):** Shows the command "go run main.go" being run in the terminal, resulting in the message "server started at localhost:9000".

Tampilan localhost pada website:



### Latihan 3: 3-ajax-multi-upload

Implementasinya (Source Code):

- main.go

File Edit Selection View Go Run ... ← → pbw

EXPLORER

PBW

- praktikum
  - pertemuan02
  - pertemuan03
  - pertemuan04
  - pertemuan05
  - pertemuan06
  - pertemuan07
  - pertemuan08 latihan
    - 1-ajax-json-payload
    - 2-ajax-json-response
    - 3-ajax-multi-upload
      - assets
      - jquery-3.3.1.min.js
    - files
- git-command.txt
- teori
- basic-go
- p06
- 00-dasar-pemrograman-golang.pdf
- README.md

main.go

```
main.go x
package main

import "fmt"
import "net/http"
import "html/template"
import "path/filepath"
import "io"
import "os"

func main() {
    http.HandleFunc("/", handleIndex)
    http.HandleFunc("/upload", handleUpload)
    http.Handle("/static/", http.StripPrefix("/static/", http.FileServer(http.Dir("assets"))))
    fmt.Println("server started at localhost:9000")
    http.ListenAndServe(":9000", nil)
}

func handleIndex(w http.ResponseWriter, r *http.Request) {
    tmpl := template.Must(template.ParseFiles("view.html"))
    if err := tmpl.Execute(w, nil); err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
    }
}

func handleUpload(w http.ResponseWriter, r *http.Request) {
    if r.Method != "POST" {
        http.Error(w, "Only accept POST request", http.StatusBadRequest)
        return
    }
    basePath := os.Getwd()
}
```

Ln 1, Col 1 Tab Size: 4 UTF-8 LF ↵ Go 1.22.1 ⚡ Go Live ⚡ Prettier

17/05/2024

File Edit Selection View Go Run ... ← → pbw

EXPLORER

PBW

- praktikum
  - pertemuan02
  - pertemuan03
  - pertemuan04
  - pertemuan05
  - pertemuan06
  - pertemuan07
  - pertemuan08 latihan
    - 1-ajax-json-payload
    - 2-ajax-json-response
    - 3-ajax-multi-upload
      - assets
      - jquery-3.3.1.min.js
    - files
- git-command.txt
- teori
- basic-go
- p06
- 00-dasar-pemrograman-golang.pdf
- README.md

main.go

```
main.go x
package main

func handleUpload(w http.ResponseWriter, r *http.Request) {
    basePath, _ := os.Getwd()

    reader, err := r.MultipartReader()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    for {
        part, err := reader.NextPart()
        if err == io.EOF {
            break
        }

        fileLocation := filepath.Join(basePath, "files", part.FileName())
        dst, err := os.Create(fileLocation)
        if dst != nil {
            defer dst.Close()
        }
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }

        if _, err := io.Copy(dst, part); err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
    }
}
```

Ln 1, Col 1 Tab Size: 4 UTF-8 LF ↵ Go 1.22.1 ⚡ Go Live ⚡ Prettier

17/05/2024



The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar contains a tree view of a project structure under the 'EXPLORER' tab. The main area displays a Go file named 'main.go' with line numbers and code completion suggestions. The status bar at the bottom shows file details like 'master' and '1708 17/07/2024'.

```
func handleUpload(w http.ResponseWriter, r *http.Request) {
    for {
        part, err := reader.NextPart()
        if err == io.EOF {
            break
        }

        fileLocation := filepath.Join(basePath, "files", part.FileName())
        dst, err := os.Create(fileLocation)
        if dst != nil {
            defer dst.Close()
        }
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }

        if _, err := io.Copy(dst, part); err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
    }
    w.Write([]byte('all files uploaded'))
}
```

- view.html

The screenshot shows a Microsoft Edge browser window with a Go Live preview of a file named 'view.html'. The browser's address bar contains 'pbw'. The preview shows an HTML page with a title 'Multiple Upload' and a script that handles multiple file uploads using jQuery and FormData.

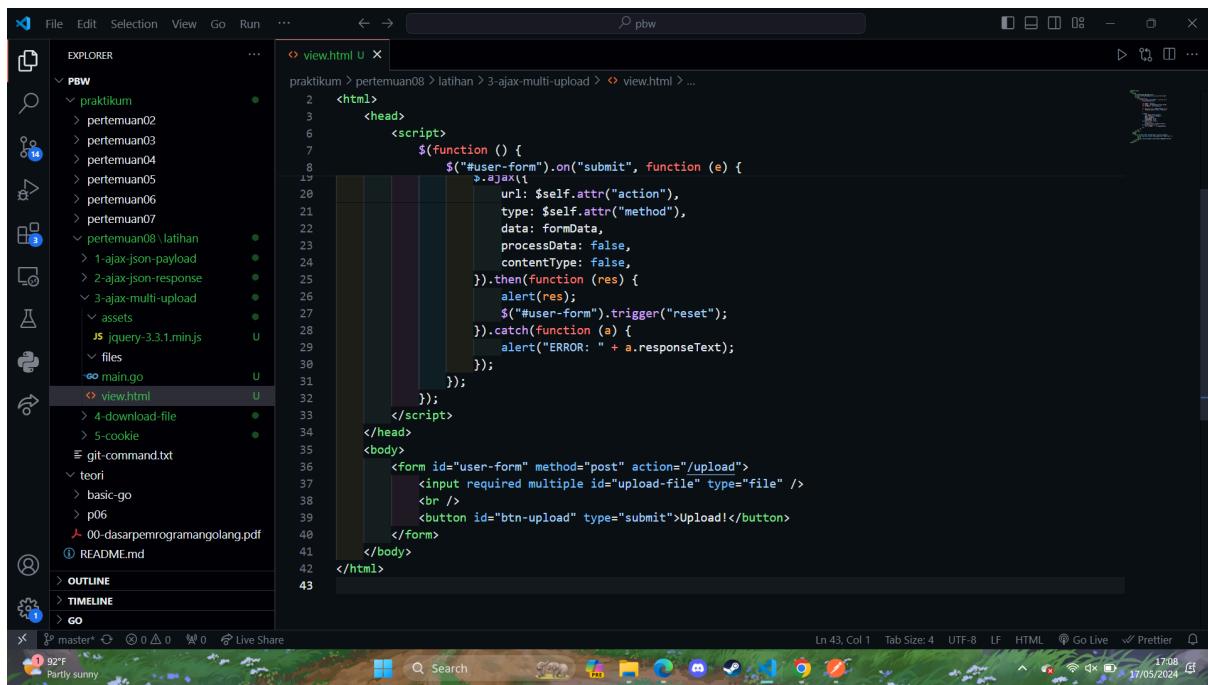
```
<!DOCTYPE html>
<html>
  <head>
    <title>Multiple Upload</title>
    <script src="static/jquery-3.3.1.min.js"></script>
    <script>
      $(function () {
        $("#user-form").on("submit", function (e) {
          e.preventDefault();

          var $self = $(this);
          var files = $("#upload-file")[0].files;
          var formData = new FormData();

          for (var i = 0; i < files.length; i++) {
            formData.append("files", files[i]);
          }

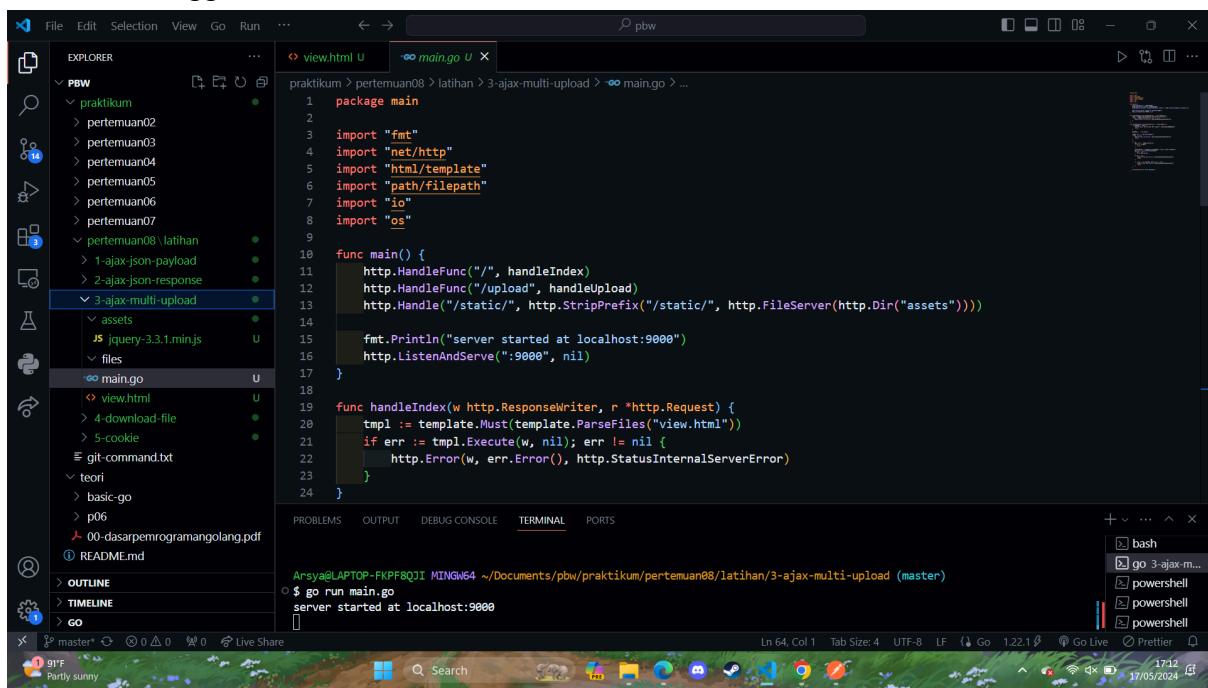
          $.ajax({
            url: $self.attr("action"),
            type: $self.attr("method"),
            data: formData,
            processData: false,
            contentType: false,
          }).then(function (res) {
            alert(res);
            $("#user-form").trigger("reset");
          }).catch(function (a) {
            alert("ERROR: " + a.responseText);
          });
        });
      });
    </script>
  </head>
  <body>
    <form id="user-form" action="upload.php" method="post">
      <input type="file" name="upload-file" multiple="multiple"/>
      <input type="submit" value="Upload" />
    </form>
  </body>
</html>
```

The browser's status bar indicates 'Ln 42, Col 8' and 'Tab Size: 4'. The taskbar at the bottom shows various pinned icons and the date '17/05/2024'.



```
<html>
  <head>
    <script>
      $(function () {
        $("#user-form").on("submit", function (e) {
          e.preventDefault();
          $.ajax({
            url: $self.attr("action"),
            type: $self.attr("method"),
            data: formData,
            processData: false,
            contentType: false,
          }).then(function (res) {
            alert(res);
            $("#user-form").trigger("reset");
          }).catch(function (a) {
            alert("ERROR: " + a.responseText);
          });
        });
      });
    </script>
  </head>
  <body>
    <form id="user-form" method="post" action="/upload">
      <input required multiple id="upload-file" type="file" />
      <br />
      <button id="btn-upload" type="submit">Upload!</button>
    </form>
  </body>
</html>
```

Run code menggunakan terminal:



```
package main
import "fmt"
import "net/http"
import "html/template"
import "path/filepath"
import "io"
import "os"

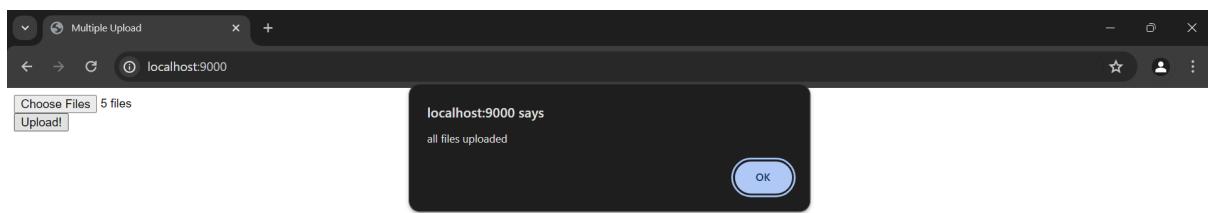
func main() {
    http.HandleFunc("/", handleIndex)
    http.HandleFunc("/upload", handleUpload)
    http.Handle("/static/", http.StripPrefix("/static/", http.FileServer(http.Dir("assets"))))
    fmt.Println("server started at localhost:9000")
    http.ListenAndServe(":9000", nil)
}

func handleIndex(w http.ResponseWriter, r *http.Request) {
    tmpl := template.Must(template.ParseFiles("view.html"))
    if err := tmpl.Execute(w, nil); err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
    }
}
```

Arsyaa@LAPTOP-FKPF8QJI MINGW64 ~\Documents\pbw\praktikum\pertemuan08\latihan\3-ajax-multi-upload (master)

```
$ go run main.go
server started at localhost:9000
```

Tampilan localhost pada website:



```
praktikum > pertemuan08 > latihan > 3-ajax-multi-upload > main.go > ...
1 package main
2
3 import "fmt"
4 import "net/http"
5 import "html/template"
6 import "path/filepath"
7 import "io"
8 import "os"
9
10 func main() {
11     http.HandleFunc("/", handleIndex)
12     http.HandleFunc("/upload", handleUpload)
13     http.Handle("/static/", http.StripPrefix("/static/", http.FileServer(http.Dir("assets"))))
14
15     fmt.Println("server started at localhost:9000")
16     http.ListenAndServe(":9000", nil)
17 }
18
19 func handleIndex(w http.ResponseWriter, r *http.Request) {
20     tmpl := template.Must(template.ParseFiles("view.html"))
21     if err := tmpl.Execute(w, nil); err != nil {
22         http.Error(w, err.Error(), http.StatusInternalServerError)
23     }
24 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Arsya@LAPTOP-FKPF8QJ1 MINGW64 ~/Documents/pbw/praktikum/pertemuan08/latihan/3-ajax-multi-upload (master)
$ go run main.go
server started at localhost:9000
```

Ln 64, Col 1 Tab Size: 4 UTF-8 LF ↴ Go 1.22.1 ⚡ Go Live ⚡ Prettier ⚡

## Latihan 4: 4-download-file

Implementasinya (Source Code):

- main.go

```
praktikum > pertemuan08 > latihan > 4-download-file > main.go > ...
1 package main
2
3 import "fmt"
4 import "net/http"
5 import "html/template"
6 import "path/filepath"
7 import "io"
8 import "encoding/json"
9 import "os"
10
11 type M map[string]interface{}
12
13 func main() {
14     http.HandleFunc("/", handleIndex)
15     http.HandleFunc("/list-files", handleListFiles)
16     http.HandleFunc("/download", handleDownload)
17
18     fmt.Println("server started at localhost:9000")
19     http.ListenAndServe(":9000", nil)
20 }
21
22 func handleIndex(w http.ResponseWriter, r *http.Request) {
23     tmpl := template.Must(template.ParseFiles("view.html"))
24     if err := tmpl.Execute(w, nil); err != nil {
25         http.Error(w, err.Error(), http.StatusInternalServerError)
26     }
27 }
28
29 func handleListFiles(w http.ResponseWriter, r *http.Request) {
30     files := []M{}
31     basePath, _ := os.Getwd()
32     filesLocation := filepath.Join(basePath, "files")
```

Ln 1, Col 1 Tab Size: 4 UTF-8 LF ↴ Go 1.22.1 ⚡ Go Live ⚡ Prettier ⚡

The screenshot shows a code editor window with the file `main.go` open. The code implements a file download handler. It uses `filepath.Walk` to traverse a directory, collects files into a slice, and then uses `json.Marshal` to convert the slice into a JSON response. The code editor interface includes a sidebar with project files like `praktikum`, `view.html`, and `README.md`. The status bar at the bottom shows the file path as `Jalan Raya Marg...`.

```
praktikum > pertemuan08 > latihan > 4-download-file > main.go > ...
func handleListFiles(w http.ResponseWriter, r *http.Request) {
    basePath, _ := os.Getwd()
    filesLocation := filepath.Join(basePath, "files")

    err := filepath.Walk(filesLocation, func(path string, info os.FileInfo, err error) error {
        if err != nil {
            return err
        }

        if info.IsDir() {
            return nil
        }

        files = append(files, M{"filename": info.Name(), "path": path})
    })
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    res, err := json.Marshal(files)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    w.Header().Set("Content-Type", "application/json")
    w.Write(res)
}
```

The screenshot shows a code editor window with the file `main.go` open. This version of the code handles file download requests. It reads the path from the request form, opens the file, and then copies its contents into the response writer. The code editor interface is similar to the first one, with a sidebar showing project files. The status bar at the bottom shows the file path as `Jalan Raya Marg...`.

```
praktikum > pertemuan08 > latihan > 4-download-file > main.go > ...
func handleListFiles(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-Type", "application/json")
    w.Write(res)
}

func handleDownload(w http.ResponseWriter, r *http.Request) {
    if err := r.ParseForm(); err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    path := r.FormValue("path")
    f, err := os.Open(path)
    if f != nil {
        defer f.Close()
    }
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    contentDisposition := fmt.Sprintf("attachment; filename=%s", f.Name())
    w.Header().Set("Content-Disposition", contentDisposition)

    if _, err := io.Copy(w, f); err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
}
```

- `view.html`

A screenshot of a code editor window titled "view.html". The code is written in JavaScript and HTML. It includes a title, a script block containing a function Yo() that handles file download logic, and another function getAlllistFiles() that uses XMLHttpRequest to get file lists. The code editor has a dark theme with syntax highlighting.

```
<!DOCTYPE html>
<html>
<head>
<title>Download file</title>
<script>
function Yo() {
    var self = this;
    var $ul = document.getElementById("list-files");

    var renderData = function (res) {
        res.forEach(function (each) {
            var $li = document.createElement("li");
            var $a = document.createElement("a");
            $li.innerText = "download ";
            $li.appendChild($a);
            $ul.appendChild($li);

            $a.href = "/download?path=" + encodeURI(each.path);
            $a.innerText = each.filename;
            $a.target = "_blank";
        });
    };

    var getAlllistFiles = function () {
        var xhr = new XMLHttpRequest();
        xhr.open("GET", "/list-files");
        xhr.onreadystatechange = function () {
            if (xhr.readyState == 4 && xhr.status == 200) {
                var json = JSON.parse(xhr.responseText);
                renderData(json);
            }
        };
    };
}

self.init = function () {
    getAlllistFiles();
};

window.onload = function () {
    new Yo().init();
};
</script>
</head>
<body>
<ul id="list-files"></ul>
</body>
</html>
```

A screenshot of a code editor window titled "view.html". The code is identical to the one in the previous screenshot, but the "renderData" and "getAlllistFiles" functions have been partially collapsed or removed, appearing as single-line blocks of code.

```
<!DOCTYPE html>
<html>
<head>
<title>Download file</title>
<script>
function Yo() {
    var self = this;
    var $ul = document.getElementById("list-files");

    var getAlllistFiles = function () {
        var xhr = new XMLHttpRequest();
        xhr.open("GET", "/list-files");
        xhr.onreadystatechange = function () {
            if (xhr.readyState == 4 && xhr.status == 200) {
                var json = JSON.parse(xhr.responseText);
                renderData(json);
            }
        };
    };
}

self.init = function () {
    getAlllistFiles();
};

window.onload = function () {
    new Yo().init();
};
</script>
</head>
<body>
<ul id="list-files"></ul>
</body>
</html>
```

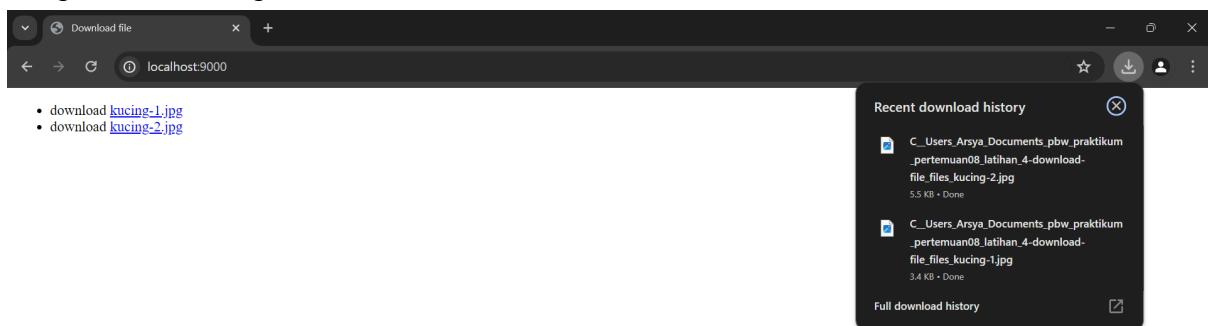
Run code menggunakan terminal:

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left displays a project structure under 'praktikum'. The main editor area shows the 'main.go' file with the following Go code:

```
1 package main
2
3 import "fmt"
4 import "net/http"
5 import "html/template"
6 import "path/filepath"
7 import "io"
8 import "encoding/json"
9 import "os"
10
11 type M map[string]interface{}
12
13 func main() {
14     http.HandleFunc("/", handleIndex)
15     http.HandleFunc("/list-files", handleListFiles)
16     http.HandleFunc("/download", handleDownload)
17
18     fmt.Println("server started at localhost:9000")
19     http.ListenAndServe(":9000", nil)
20 }
21
22 func handleIndex(w http.ResponseWriter, r *http.Request) {
23     tmpl := template.Must(template.ParseFiles("view.html"))
24     if err := tmpl.Execute(w, nil); err != nil {
```

The terminal tab at the bottom shows the command \$ go run main.go followed by the output server started at localhost:9000.

Tampilan localhost pada website:



## Latihan 5: 5-cookie

Implementasinya (Source Code):

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure under "praktikum" named "pertemuan08". Inside "pertemuan08", there are several subfolders like "latihan", "teori", and "basic-go". The "main.go" file is selected.
- Editor:** The main editor tab displays the "main.go" file content. The code implements a simple HTTP server that handles cookie creation and deletion. It uses the standard library's "net/http" package and "fmt" for output.
- Status Bar:** Shows the current line (Ln 32, Col 1), tab size (Tab Size: 4), and file encoding (UTF-8).
- Bottom:** The taskbar includes icons for various applications like Microsoft Word, Excel, and Google Chrome.

```
package main

import (
    "fmt"
    "net/http"
    "time"
)

gubrak "github.com/novalagung/gubrak/v2"

type M map[string]interface{}

var cookieName = "CookieData"

func ActionIndex(w http.ResponseWriter, r *http.Request) {
    c := &http.Cookie{}

    if storedCookie, _ := r.Cookie(cookieName); storedCookie != nil {
        c = storedCookie
    }

    if c.Value == "" {
        c = &http.Cookie{}
        c.Name = cookieName
        c.Value = gubrak.RandomString(32)
        c.Expires = time.Now().Add(5 * time.Minute)
        http.SetCookie(w, c)
    }

    w.Write([]byte(c.Value))
}

func ActionDelete(w http.ResponseWriter, r *http.Request) {
    c := &http.Cookie{}
    c.Name = cookieName
    c.Expires = time.Unix(0, 0)
    c.MaxAge = -1
    http.SetCookie(w, c)

    http.Redirect(w, r, "/", http.StatusTemporaryRedirect)
}

func main() {
    http.HandleFunc("/", ActionIndex)
    http.HandleFunc("/delete", ActionDelete)

    fmt.Println("server started at localhost:9000")
    http.ListenAndServe(":9000", nil)
}
```

This screenshot shows the same VS Code environment after adding more code to the "main.go" file. The new code defines a function "ActionDelete" which sets a cookie with an expiration time of zero, effectively deleting it. It also adds a redirect to the root URL. The "main" function now handles both "/index" and "/delete" routes.

```
func ActionIndex(w http.ResponseWriter, r *http.Request) {
    c := &http.Cookie{}

    if c.Value == "" {
        c = &http.Cookie{}
        c.Name = cookieName
        c.Value = gubrak.RandomString(32)
        c.Expires = time.Now().Add(5 * time.Minute)
        http.SetCookie(w, c)
    }

    w.Write([]byte(c.Value))
}

func ActionDelete(w http.ResponseWriter, r *http.Request) {
    c := &http.Cookie{}
    c.Name = cookieName
    c.Expires = time.Unix(0, 0)
    c.MaxAge = -1
    http.SetCookie(w, c)

    http.Redirect(w, r, "/", http.StatusTemporaryRedirect)
}

func main() {
    http.HandleFunc("/", ActionIndex)
    http.HandleFunc("/delete", ActionDelete)

    fmt.Println("server started at localhost:9000")
    http.ListenAndServe(":9000", nil)
}
```

Run code menggunakan terminal:

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a file tree with a folder named 'praktikum' containing subfolders like 'pertemuan02' through 'pertemuan07', and files such as 'main.go', 'go.mod', 'go.sum', and 'git-command.txt'. The main code editor window displays a Go file named 'main.go' with the following code:

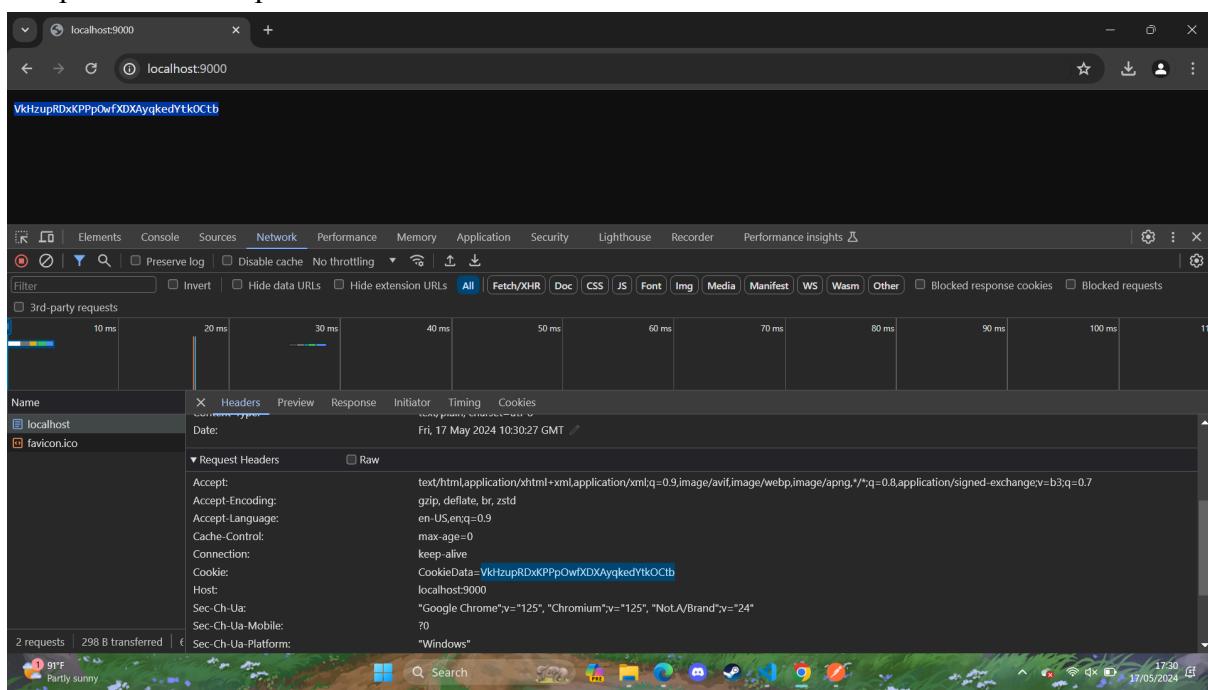
```
1 package main
2
3 import (
4     "fmt"
5     "net/http"
6     "time"
7 )
8
9 func ActionIndex(w http.ResponseWriter, r *http.Request) {
10
11     c := &http.Cookie{
12         Name: "CookieData"
13     }
14
15     if storedCookie, _ := r.Cookie("CookieData"); storedCookie != nil {
16         c = storedCookie
17     }
18
19     http.SetCookie(w, c)
20
21 }
```

The terminal tab at the bottom shows the command line output:

```
go mod init example.com/m/v2' to initialize a v2 module
Run 'go help mod init' for more information.

Arasya@LAPTOP-FKPF8QJ1 MINGW64 ~/Documents/pbw/praktikum/pertemuan08/latihan/5-cookie (master)
$ go run main.go
server started at localhost:9000
```

Tampilan localhost pada website:



## Pertanyaan:

### a. Apa itu AJAX?

AJAX (Asynchronous JavaScript and XML) merupakan teknik yang digunakan untuk membuat permintaan HTTP ke server dan menerima respon dari server tanpa harus merefresh seluruh halaman web. Teknik ini digunakan aplikasi web untuk memperbarui konten secara dinamis, sehingga meningkatkan interaktivitas dan pengalaman pengguna.

**b. Apa itu JSON? Apa kegunaannya?**

JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan dan mudah dibaca serta ditulis oleh manusia. JSON juga mudah untuk diurai (parse) dan dibentuk (generate) oleh mesin. JSON sering digunakan dalam komunikasi data antara server dan aplikasi web (terutama AJAX), karena kesederhanaannya dan kemampuannya untuk merepresentasikan struktur data kompleks seperti objek dan array.

**c. Apa itu ParseMultipartForm dan MultipartReader ? Jelaskan perbedaannya!**

- ParseMultipartForm: Fungsi ini adalah bagian dari struct http.Request yang memarsing seluruh body dari permintaan multipart/form-data dan menyimpannya dalam memori atau disk, tergantung pada batasan ukuran yang diberikan.
- MultipartReader: Ini adalah objek reader yang membaca data multipart secara streaming. Ini digunakan ketika kita ingin memproses data multipart dalam bagian-bagian kecil, yang bisa lebih efisien dalam hal penggunaan memory, terutama untuk file berukuran besar.

**d. Apa itu JavaScript? Jelaskan kegunaannya dalam frontend !**

JavaScript adalah bahasa pemrograman yang digunakan untuk membuat halaman web interaktif. Dalam konteks frontend, JavaScript memungkinkan pengembang untuk menambahkan logika dinamis ke dalam situs web, seperti merespons tindakan pengguna (klik, input, dll.), memodifikasi konten halaman secara dinamis, dan berkomunikasi dengan server tanpa merefresh halaman (menggunakan AJAX).

**e. Jelaskan apa itu EventHandler!**

EventHandler adalah fungsi atau metode yang dijalankan sebagai respons terhadap event tertentu. Dalam konteks pemrograman web, event handler digunakan untuk menangani berbagai jenis event yang dipicu oleh pengguna, seperti klik mouse, penekanan tombol, perubahan nilai input, dan lain-lain.

## Upload Ke Git

```
Arsya@LAPTOP-FKPF8QJI MINGW64 ~/Documents/pbw/praktikum/pertemuan08 (master)
$ cd latihan/
Arsa@LAPTOP-FKPF8QJI MINGW64 ~/Documents/pbw/praktikum/pertemuan08/latihan (master)
$ git add .
warning: in the working copy of 'praktikum/pertemuan08/latihan/1-ajax-json-payload/assets/jquery-3.7.1.min.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/1-ajax-json-payload/main.go', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/1-ajax-json-payload/view.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/2-ajax-json-response/main.go', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/3-ajax-multi-upload/assets/jquery-3.3.1.min.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/3-ajax-multi-upload/main.go', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/3-ajax-multi-upload/view.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/4-download-file/main.go', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/4-download-file/view.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'praktikum/pertemuan08/latihan/5-cookie/main.go', LF will be replaced by CRLF the next time Git touches it
```

```
Arsya@LAPTOP-FKPF8QJI MINGW64 ~/Documents/pbw/praktikum/pertemuan08/latihan (master)
$ git commit -m "upload latihan praktikum ke g"
[master 21a0e48] upload latihan praktikum ke g
19 files changed, 462 insertions(+)
create mode 100644 praktikum/pertemuan08/latihan/1-ajax-json-payload/assets/jquery-3.7.1.min.js
create mode 100644 praktikum/pertemuan08/latihan/1-ajax-json-payload/main.go
create mode 100644 praktikum/pertemuan08/latihan/1-ajax-json-payload/view.html
create mode 100644 praktikum/pertemuan08/latihan/2-ajax-json-response/main.go
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/assets/jquery-3.3.1.min.js
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/files/kucing-1.jpg
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/files/kucing-2.jpg
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/files/kucing-3.jpg
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/files/kucing-4.jpg
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/files/kucing-5.jpg
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/main.go
create mode 100644 praktikum/pertemuan08/latihan/3-ajax-multi-upload/view.html
create mode 100644 praktikum/pertemuan08/latihan/4-download-file/files/kucing-1.jpg
create mode 100644 praktikum/pertemuan08/latihan/4-download-file/files/kucing-2.jpg
create mode 100644 praktikum/pertemuan08/latihan/4-download-file/main.go
create mode 100644 praktikum/pertemuan08/latihan/5-cookie/go.mod
create mode 100644 praktikum/pertemuan08/latihan/5-cookie/go.sum
create mode 100644 praktikum/pertemuan08/latihan/5-cookie/main.go

Arsya@LAPTOP-FKPF8QJI MINGW64 ~/Documents/pbw/praktikum/pertemuan08/latihan (master)
$ git push -f origin master
Enumerating objects: 33, done.
Counting objects: 100%, (33/33), done.
Delta compression using up to 12 threads
Compressing objects: 100% (27/27), done.
Writing objects: 100% (31/31), 89.69 KiB | 5.98 MiB/s, done.
Total 31 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To https://github.com/Arsyad11/pbw.git
21975e4..21a0e48 master -> master
```

## Kesimpulan

Pada praktikum ini mempelajari penggunaan AJAX untuk mengirim dan menerima JSON, mengunggah beberapa file sekaligus, mengunduh file dari server, dan mengelola cookie HTTP. Latihan ini menunjukkan bagaimana AJAX dapat meningkatkan interaktivitas aplikasi web tanpa perlu memuat ulang halaman, memudahkan pertukaran data dengan format JSON, serta mengoptimalkan pengolahan data multipart untuk file upload. Selain itu, kami mempelajari pengunduhan file dari server dan manajemen sesi pengguna melalui cookie, yang semuanya krusial untuk pengembangan web yang dinamis dan responsif.

**Link Repository Github:**

<https://github.com/Arsyayd11/pbw/tree/master/praktikum/pertemuan08/latihan>