LATIHAN PRAKTIKUM
PEMROGRAMAN BERBASIS WEB (A)

DI SUSUN OLEH :

NAMA : DAFFA ABRAAR SAJUTI

NPM : 4522210040

S1- Teknik Informatika

Fakultas Teknik Universitas Pancasila

2023/2024

API MENGGUNAKAN GORILLA

1. Membuat Modul

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\daffa\Kampus\SEMESTER-4\PRAKTIKUM-PEMROGRAMAN-BERBASIS-WEB\Praktikum10_PBW> go mod init Praktikum10_PBW
go: creating new go.mod: module Praktikum10_PBW
```

2. Menginstal Library GORILLA dan UUID

```
PS C:\Users\daffa\Kampus\SEMESTER-4\PRAKTIKUM-PEMROGRAMAN-BERBASIS-WEB\Praktikum10_PBW> go get -u github.com/gorilla/mux
go: downloading github.com/gorilla/mux v1.8.1
go: added github.com/gorilla/mux v1.8.1
PS C:\Users\daffa\Kampus\SEMESTER-4\PRAKTIKUM-PEMROGRAMAN-BERBASIS-WEB\Praktikum10_PBW> go get github.com/google/uuid
go: downloading github.com/google/uuid v1.6.0
```

- **GORILLA**

  Gorilla adalah sekumpulan paket untuk digunakan untuk mempermudah pengembangan aplikasi web. Gorilla terdiri dari beberapa modul yang dapat digunakan secara terpisah atau bersama-sama untuk menangani berbagai aspek pengembangan web. Beberapa paket yang populer dalam Gorilla adalah:

  1. **Gorilla Mux**: Sebuah router yang sangat populer dan kuat untuk menangani rute HTTP. Mux memungkinkan penanganan rute yang kompleks dan mendukung metode HTTP, variabel di URL, dan middleware.
  2. **Gorilla Sessions**: Paket untuk mengelola sesi di aplikasi web. Ini memungkinkan penyimpanan data sesi di berbagai backend seperti cookie, file, memcached, dan database.
  3. **Gorilla WebSocket**: Implementasi WebSocket yang digunakan untuk komunikasi dua arah antara server dan client.
  4. **Gorilla Context**: Digunakan untuk menyimpan data kunci/ nilai selama request hidup. Ini berguna untuk berbagi data antara middleware dan handler.

- **UUID**

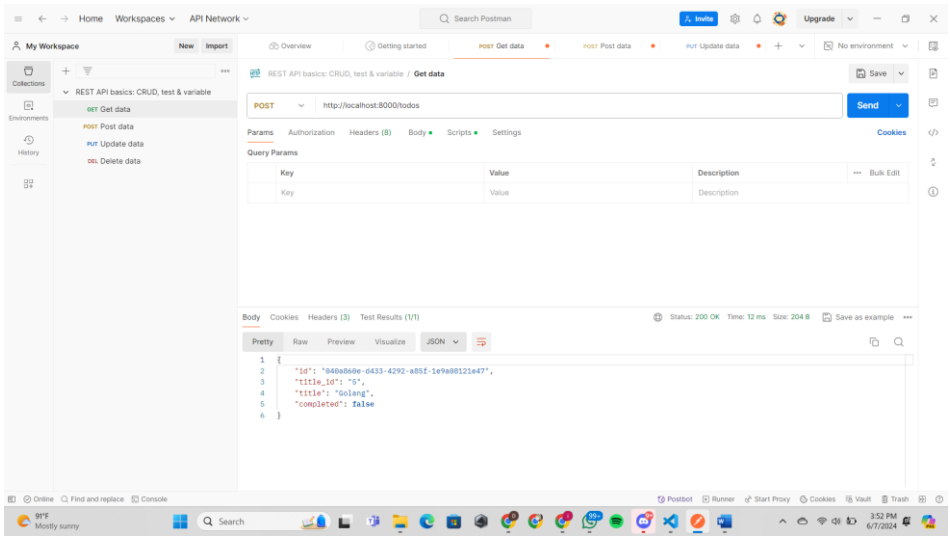
  UUID (Universally Unique Identifier) adalah sebuah standar untuk pengidentifikasi unik yang dapat digunakan di berbagai sistem komputasi. Dalam bahasa Go (Golang), UUID sering digunakan untuk membuat pengidentifikasi unik yang tidak bergantung pada waktu, lokasi, atau sistem.

3. Membuat File main.go

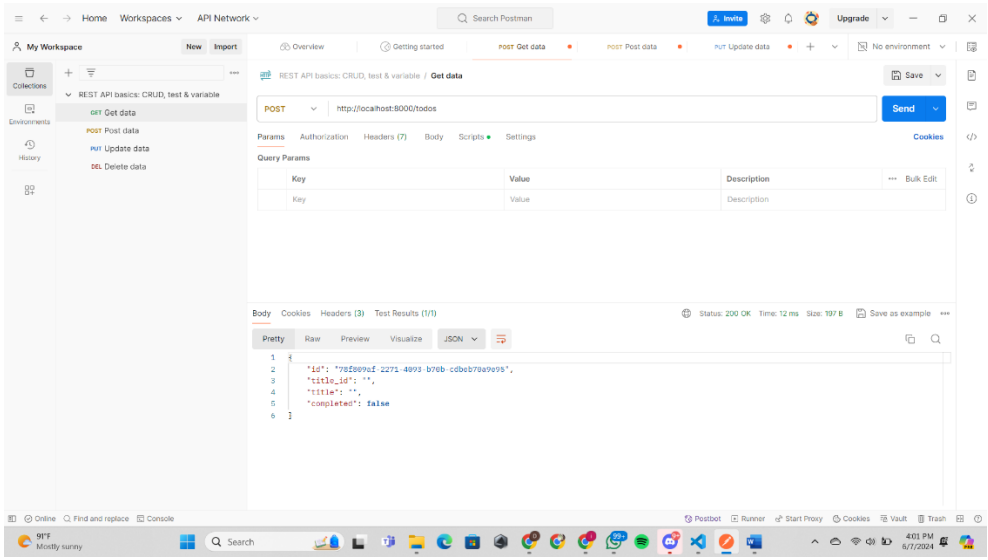
```go
package main

import (
    "encoding/json"
    "log"
    "net/http"

    "github.com/google/uuid"
    "github.com/gorilla/mux"
)

//model data
type Todo struct {
    ID        string `json:"id"`
    TitleID   string `json:"title_id"`
    Title     string `json:"title"`
    Completed bool   `json:"completed"`
}

// inisialisasi slice untuk menyimpan data
var todos []Todo

func main() {
    // Inisialisasi router
    router := mux.NewRouter()

    // Menambahkan handler untuk route "/todos"
    router.HandleFunc("/todos", GetTodos).Methods("GET")
    router.HandleFunc("/todos", CreateTodo).Methods("POST")

    // Menjalankan server pada port 8000
    log.Fatal(http.ListenAndServe(":8000", router))
}

// Handler untuk mendapatkan semua todos
func GetTodos(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-Type", "application/json")
    json.NewEncoder(w) .Encode(todos)
}

// Handler untuk membuat todo baru
func CreateTodo(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-Type", "application/json")
    var newTodo Todo
    json. NewDecoder(r.Body) .Decode(&newTodo)
    newTodo.ID = uuid.New().String()
    todos = append(todos, newTodo)
    json. NewEncoder(w) . Encode(newTodo)
}
```

4. Menjalankan file main.go dan membuka postman

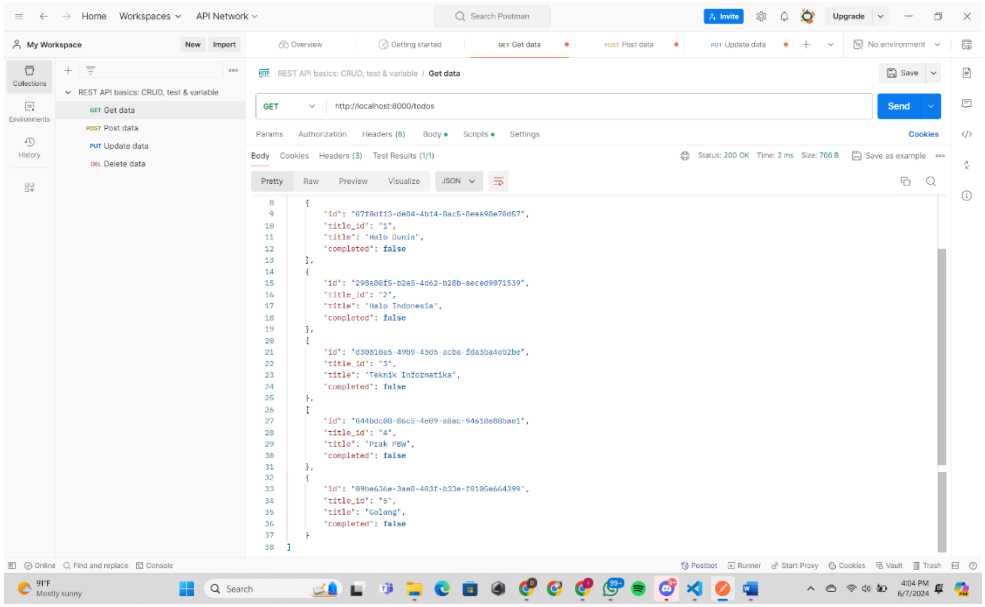
```
PS C:\Users\daffa\Kampus\SEMESTER-4\PRAKTIKUM-PEMROGRAMAN-BERBASIS-WEB\Praktikum10_PBW> go run .\main.go
```



5. **Memasukkan title dengan body JSON dengan metode POST**



6. **Melihat title yang sudah di buat menggunakan metode GET**



7. **Link GitHub**

   https://github.com/DaffaAbraarSajuti/Praktikum-Pemrograman-Berbasis-Web/tree/master/Praktikum10_PBW