ACT

Mata Praktikum	: GOLANG INTERMEDIET
Kelas	: 3IA20
Praktikum ke-	: 7
Tanggal	: 4 November 2022
Materi	: (CRUD MYSQL PADA GOLANG)
NPM	: 50420900
Nama	: Muhammad Reza Hidayat
Ketua Asisten	:
Nama Asisten	:
Paraf Asisten	:
Jumlah Lembar	:

LABORATORIUM TEKNIK INFORMATIKA

UNIVERSITAS GUNADARMA

2022

A. LISTING PROGRAM

1. Apa yang dimaksud dengan crud? (10 poin)

Jawab :

CRUD merupakan istilah yang digunakan dalam pembuatan atau pengolahan struktur basis data. CRUD terdiri dari perintah Create, Read, Update, dan Delete.

- Create => merupakan struktur pertama pada CRUD yang berfungsi untuk membuat record baru pada sistem basis data. Dengan perintah ini memungkinkan administrator untuk membuat database atau table baru.
- Read => merupakan perintah yang digunakan untuk mengambil atau mencari data tertentu datalam suatu database atau table. Fungsi read kurang lebih memiliki fungsi yang sama dengan perintah select ataupun search.
- Update => merupakan fungsi yang sering diguanakan untuk melakukan perubahan pada data dalam suatu database. Perintah update adalah perintah yang digunakan untuk melakukan modifikasi pada adata yang sudah disipan sebelumnya.
- Delete => merupakan perintah yang biasa digunakan untuk melakukan penghapusan terhadap suatu data. Ketika suatu data sudah tidak diperlukan atau terjadi kesalahan dalam penginputan, maka perintah ini dapat digunakan untuk mengatasinya.

2. Sebutkan struktur crud pada Go Language! (10 poin)

Struktur Create

reza_50420900_pert7 - main.go

```go
func createMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var response ResponseMessage

    db := dbConn()
    defer db.Close()

    err := r.ParseForm()
    if err != nil {
        log.Print(err.Error())
   }

    npm := r.Form.Get("npm")
    nama := r.Form.Get("nama")
    kelas := r.Form.Get("kelas")
    profile := "gambar1.jpg"

    rows, err := db.Prepare("INSERT INTO mahasiswa(npm, nama, kelas, profile) VALUES(?, ?, ?, ?)")

    if err != nil {
        log.Print(err.Error())
   }

    rows.Exec(npm, nama, kelas, profile)

    response.Status = true
    response.Message = "Mahasiswa berhasil ditambahkan"

    log.Print(response.Message)

    json.NewEncoder(w).Encode(response)
}
```

Struktur Update

reza_50420900_pert7 - main.go

```go
func updateMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var response ResponseMessage
    var responseErr ResponseError
    var mahasiswa Mahasiswa

    db := dbConn()
    defer db.Close()

    err := r.ParseForm()
    if err != nil {
        log.Print(err.Error())
    }

    id := r.Form.Get("id")
    npm := r.Form.Get("npm")
    nama := r.Form.Get("nama")
    kelas := r.Form.Get("kelas")

    rows := db.QueryRow("SELECT id FROM mahasiswa WHERE id=?", id)
    if err := rows.Scan(&mahasiswa.Id); err != nil && err == sql.ErrNoRows {
        responseErr.Status = false
        responseErr.Error = "Mahasiswa tidak ditemukan"
        w.WriteHeader(http.StatusNotFound)
        json.NewEncoder(w).Encode(responseErr)
        return
    }

    update, err := db.Prepare("UPDATE mahasiswa SET npm=?, nama=?, kelas=? WHERE id=?")

    if err != nil {
        log.Print(err.Error())
    }

    update.Exec(npm, nama, kelas, id)
    response.Status = true
    response.Message = "Data mahasiswa berhasil diubah"

    log.Print(response.Message)

    json.NewEncoder(w).Encode(response)
    return
}
```

Struktur Delete

reza_50420900_pert7 - main.go

```go
func deleteMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var mahasiswa Mahasiswa
    var response ResponseMessage
    var responseErr ResponseError

    db := dbConn()
    defer db.Close()

    params := mux.Vars(r)

    rows := db.QueryRow("SELECT id FROM mahasiswa WHERE id=?", params["id"])
    if err := rows.Scan(&mahasiswa.Id); err != nil && err == sql.ErrNoRows {
        responseErr.Status = false
        responseErr.Error = "Mahasiswa tidak ditemukan"
        w.WriteHeader(http.StatusNotFound)
        json.NewEncoder(w).Encode(responseErr)
        return
    }

    delete, err := db.Prepare("DELETE FROM mahasiswa WHERE id=?")

    if err != nil {
        log.Print(err.Error())
    }

    delete.Exec(params["id"])

    response.Status = true
    response.Message = "Data mahasiswa berhasil dihapus"

    log.Print(response.Message)

    json.NewEncoder(w).Encode(response)
    return
}
```

reza_50420900_pert7 - main.go

```go
func main() {
    r := mux.NewRouter().StrictSlash(true)
    mime.AddExtensionType(".js", "application/javascript")

    r.HandleFunc("/api/mahasiswa", getAllMahasiswa).Methods("GET")
    r.HandleFunc("/api/mahasiswa/{id}", getMahasiswa).Methods("GET")
    r.HandleFunc("/api/mahasiswa", createMahasiswa).Methods("POST")
    r.HandleFunc("/api/mahasiswa", updateMahasiswa).Methods("PUT")
    r.HandleFunc("/api/mahasiswa/{id}", deleteMahasiswa).Methods("DELETE")

    r.HandleFunc("/mahasiswa/search/{keyword}", getMahasiswaByName).Methods("GET")

    spa := spaHandler{staticPath: "polymer", indexPath: "index.html"}
    r.PathPrefix("/").Handler(spa)

    srv := &http.Server{
        Handler:      r,
        Addr:         "127.0.0.1:8000", // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
        WriteTimeout: 15 * time.Second,
        ReadTimeout:  15 * time.Second,
    }

    log.Print("Server berjalan di http://127.0.0.1:8000") // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
    srv.ListenAndServe()
}
```

3. Membuat dan menginput data pada table database! (20 poin)

SQL COMMAND

undefined - pertemuan7.sql

```sql
--Membuat database reza_50420900_pert7
CREATE DATABASE reza_50420900_pert7;

--menggunakan database reza_50420900_pert7
use reza_50420900_pert7;

--membuat tabel mahasiswa
CREATE TABLE mahasiswa (
    id int(6) unsigned AUTO_INCREMENT primary key,
    npm char(8) NOT NULL,
    nama varchar(30) NOT NULL,
    kelas char(5) NOT NULL,
    profile varchar(30)NOT NULL);

--menampilkan tabel employee
select * from mahasiswa;

--menampilkan database employee
desc mahasiswa;
```

Main.go

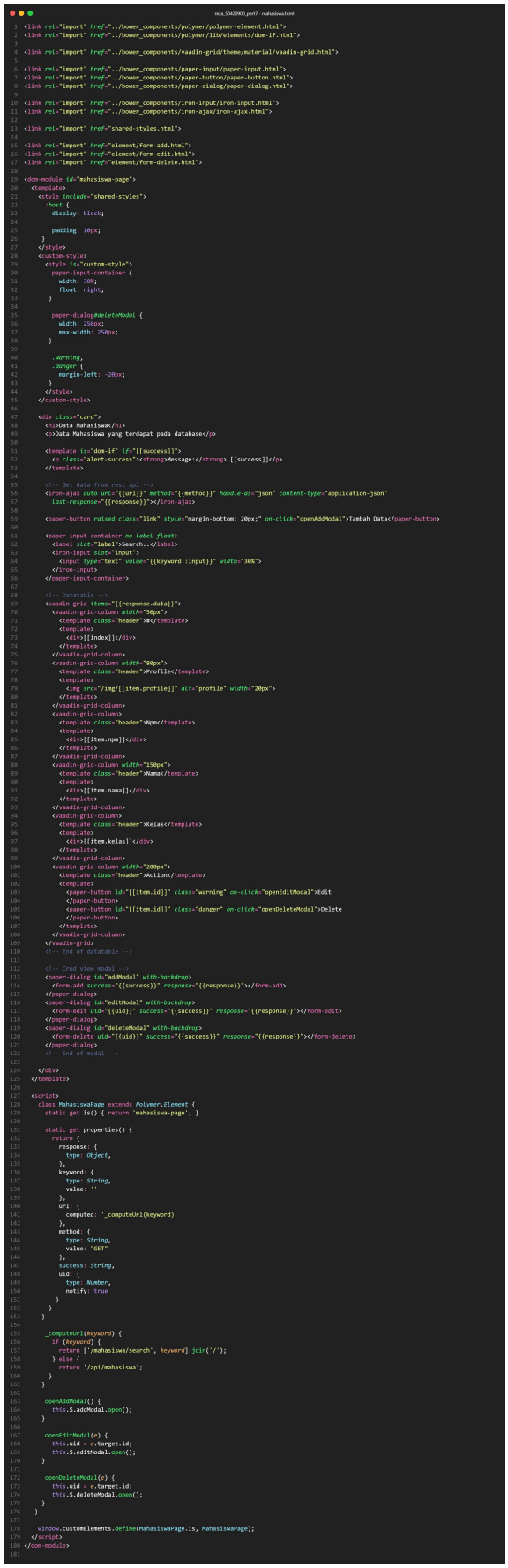
reza_50420900_pert7 - main.go

```go
package main

import (
    "database/sql"
    "encoding/json"
    "fmt"
    "log"
    "mime"
    "net/http"
    "os"
    "path/filepath"
    "time"

    _ "github.com/go-sql-driver/mysql"
    "github.com/gorilla/mux"
)

type spaHandler struct {
    staticPath string
    indexPath  string
}

type Mahasiswa struct {
    Id      int    `json:"id"`
    Npm     string `json:"npm"`
    Nama    string `json:"nama"`
    Kelas   string `json:"kelas"`
    Profile string `json:"profile"`
}

type ResponseAllData struct {
    Status bool        `json:"status"`
    Data   []Mahasiswa `json:"data"`
}

type ResponseData struct {
    Status bool      `json:"status"`
    Data   Mahasiswa `json:"data"`
}

type ResponseMessage struct {
    Status  bool   `json:"status"`
    Message string `json:"message"`
}

type ResponseError struct {
    Status bool   `json:"status"`
    Error  string `json:"error"`
}

func dbConn() (db *sql.DB) {
    dbDriver := "mysql"
    dbName := "reza_50420900_pert7" // Diganti menjadi nama database kalian masing-masing
    dbUser := "root"
    dbPass := ""
    db, err := sql.Open(dbDriver, dbUser+":"+dbPass+"@tcp(localhost)/"+dbName)

    if err != nil {
        panic(err.Error())
    }

    return db
}

func getAllMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var response ResponseAllData
    var mahasiswa Mahasiswa
    var mhs []Mahasiswa

    db := dbConn()
    rows, err := db.Query("SELECT * FROM mahasiswa")
    defer db.Close()

    if err != nil {
        log.Print(err.Error())
    }

    for rows.Next() {
        err := rows.Scan(&mahasiswa.Id, &mahasiswa.Npm, &mahasiswa.Nama, &mahasiswa.Kelas, &mahasiswa.Profile)

        if err != nil {
            log.Print(err.Error())
        } else {
            mhs = append(mhs, mahasiswa)
        }
    }

    response.Status = true
    response.Data = mhs

    json.NewEncoder(w).Encode(response)
    return
}

func getMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var response ResponseData
    var responseErr ResponseError
    var mahasiswa Mahasiswa

    db := dbConn()
    defer db.Close()

    params := mux.Vars(r)

    rows := db.QueryRow("SELECT * FROM mahasiswa WHERE id=?", params["id"])
    err := rows.Scan(&mahasiswa.Id, &mahasiswa.Npm, &mahasiswa.Nama, &mahasiswa.Kelas, &mahasiswa.Profile)

    if err != nil && err == sql.ErrNoRows {
        responseErr.Status = false
        responseErr.Error = "Mahasiswa tidak ditemukan"
        w.WriteHeader(http.StatusNotFound)
        json.NewEncoder(w).Encode(responseErr)
        return
    }

    response.Status = true
    response.Data = mahasiswa

    json.NewEncoder(w).Encode(response)
    return
}

func getMahasiswaByName(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var mahasiswa Mahasiswa
    var mhs []Mahasiswa
    var response ResponseAllData

    db := dbConn()
    defer db.Close()

    params := mux.Vars(r)
    query := fmt.Sprintf("SELECT * FROM mahasiswa WHERE nama LIKE '%%%%'", params["keyword"])

    rows, err := db.Query(query)

    if err != nil {
        log.Print(err.Error())
    }

    for rows.Next() {
        if err := rows.Scan(&mahasiswa.Id, &mahasiswa.Npm, &mahasiswa.Nama, &mahasiswa.Kelas, &mahasiswa.Profile); err != nil {
            log.Print(err.Error())
        }

        mhs = append(mhs, mahasiswa)
    }

    response.Status = true
    response.Data = mhs

    json.NewEncoder(w).Encode(response)
    return
}
```

reza_50420900_pert7 - main.go

```go
func createMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var response ResponseMessage

    db := dbConn()
    defer db.Close()

    err := r.ParseForm()
    if err != nil {
        log.Print(err.Error())
    }

    npm := r.Form.Get("npm")
    nama := r.Form.Get("nama")
    kelas := r.Form.Get("kelas")
    profile := "gambar1.jpg"

    rows, err := db.Prepare("INSERT INTO mahasiswa(npm, nama, kelas, profile) VALUES(?, ?, ?, ?)")

    if err != nil {
        log.Print(err.Error())
    }

    rows.Exec(npm, nama, kelas, profile)

    response.Status = true
    response.Message = "Mahasiswa berhasil ditambahkan"

    log.Print(response.Message)

    json.NewEncoder(w).Encode(response)
}

func updateMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var response ResponseMessage
    var responseErr ResponseError
    var mahasiswa Mahasiswa

    db := dbConn()
    defer db.Close()

    err := r.ParseForm()
    if err != nil {
        log.Print(err.Error())
    }

    id := r.Form.Get("id")
    npm := r.Form.Get("npm")
    nama := r.Form.Get("nama")
    kelas := r.Form.Get("kelas")

    rows := db.QueryRow("SELECT id FROM mahasiswa WHERE id=?", id)
    if err := rows.Scan(&mahasiswa.Id); err != nil && err == sql.ErrNoRows {
        responseErr.Status = false
        responseErr.Error = "Mahasiswa tidak ditemukan"
        w.WriteHeader(http.StatusNotFound)
        json.NewEncoder(w).Encode(responseErr)
        return
    }

    update, err := db.Prepare("UPDATE mahasiswa SET npm=?, nama=?, kelas=? WHERE id=?")

    if err != nil {
        log.Print(err.Error())
    }

    update.Exec(npm, nama, kelas, id)
    response.Status = true
    response.Message = "Data mahasiswa berhasil diubah"

    log.Print(response.Message)

    json.NewEncoder(w).Encode(response)
    return
}

func deleteMahasiswa(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-type", "application/json")

    var mahasiswa Mahasiswa
    var response ResponseMessage
    var responseErr ResponseError

    db := dbConn()
    defer db.Close()

    params := mux.Vars(r)

    rows := db.QueryRow("SELECT id FROM mahasiswa WHERE id=?", params["id"])
    if err := rows.Scan(&mahasiswa.Id); err != nil && err == sql.ErrNoRows {
        responseErr.Status = false
        responseErr.Error = "Mahasiswa tidak ditemukan"
        w.WriteHeader(http.StatusNotFound)
        json.NewEncoder(w).Encode(responseErr)
        return
    }

    delete, err := db.Prepare("DELETE FROM mahasiswa WHERE id=?")

    if err != nil {
        log.Print(err.Error())
    }

    delete.Exec(params["id"])

    response.Status = true
    response.Message = "Data mahasiswa berhasil dihapus"

    log.Print(response.Message)

    json.NewEncoder(w).Encode(response)
    return
}

func (h spaHandler) ServeHTTP(w http.ResponseWriter, r *http.Request) {
    path := filepath.Join(h.staticPath, r.URL.Path)

    _, err := os.Stat(path)
    if os.IsNotExist(err) {
        http.ServeFile(w, r, filepath.Join(h.staticPath, h.indexPath))
        return
    } else if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }

    http.FileServer(http.Dir(h.staticPath)).ServeHTTP(w, r)
}

func main() {
    r := mux.NewRouter().StrictSlash(true)
    mime.AddExtensionType(".js", "application/javascript")

    r.HandleFunc("/api/mahasiswa", getAllMahasiswa).Methods("GET")
    r.HandleFunc("/api/mahasiswa/{id}", getMahasiswa).Methods("GET")
    r.HandleFunc("/api/mahasiswa", createMahasiswa).Methods("POST")
    r.HandleFunc("/api/mahasiswa", updateMahasiswa).Methods("PUT")
    r.HandleFunc("/api/mahasiswa/{id}", deleteMahasiswa).Methods("DELETE")

    r.HandleFunc("/mahasiswa/search/{keyword}", getMahasiswaByName).Methods("GET")

    spa := spaHandler{staticPath: "polymer", indexPath: "index.html"}
    r.PathPrefix("/").Handler(spa)

    srv := &http.Server{
        Handler:      r,
        Addr:         "127.0.0.1:8000", // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
        WriteTimeout: 15 * time.Second,
        ReadTimeout:  15 * time.Second,
    }

    log.Print("Server berjalan di http://127.0.0.1:8000") // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
    srv.ListenAndServe()
}
```

Mahasiswa.html

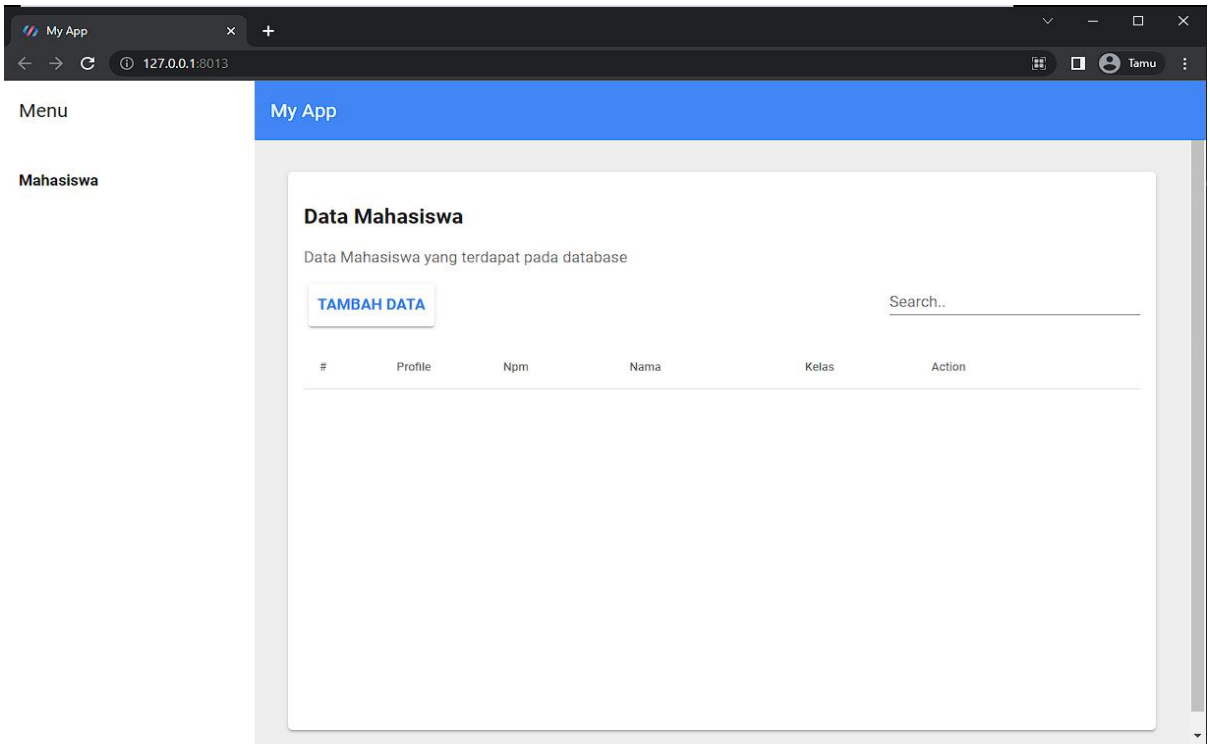


B. OUPUT PROGRAM

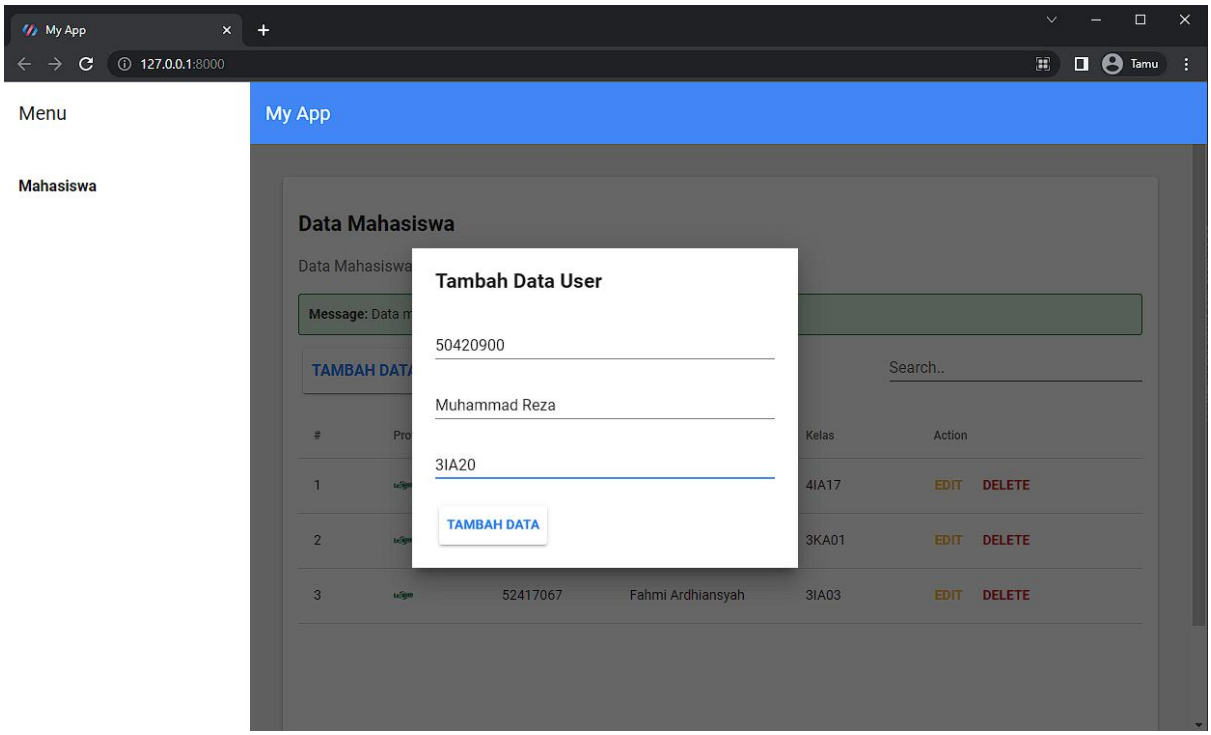
Go run main.go

```
reza_50420900_pert7 - main.go

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Program Files\Go\src\reza_50420900_pert7> go run main.go
2022/11/04 18:11:06 Server berjalan di http://127.0.0.1:8000
2022/11/04 18:12:48 Mahasiswa berhasil ditambahkan
2022/11/04 18:13:11 Mahasiswa berhasil ditambahkan
2022/11/04 18:13:29 Mahasiswa berhasil ditambahkan
2022/11/04 18:14:22 Mahasiswa berhasil ditambahkan
2022/11/04 18:15:21 Data mahasiswa berhasil diubah
2022/11/04 18:16:48 Data mahasiswa berhasil dihapus
```

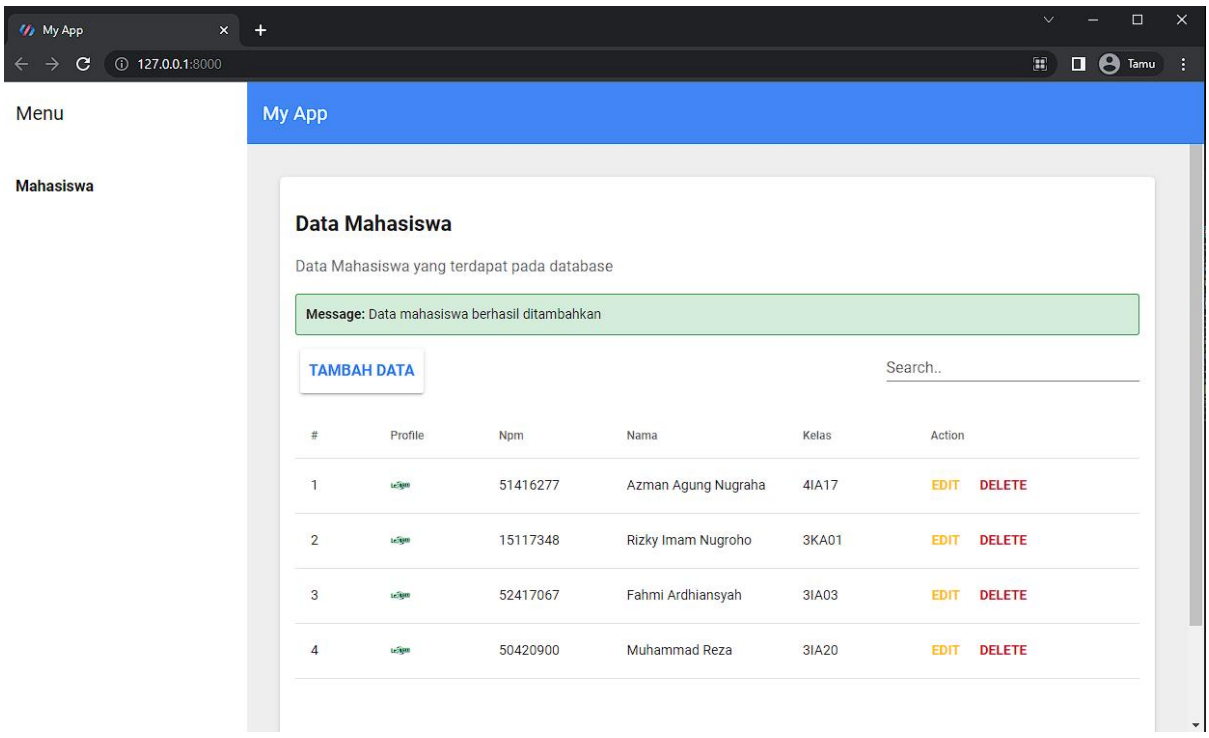
http://127.0.0.1:8000/



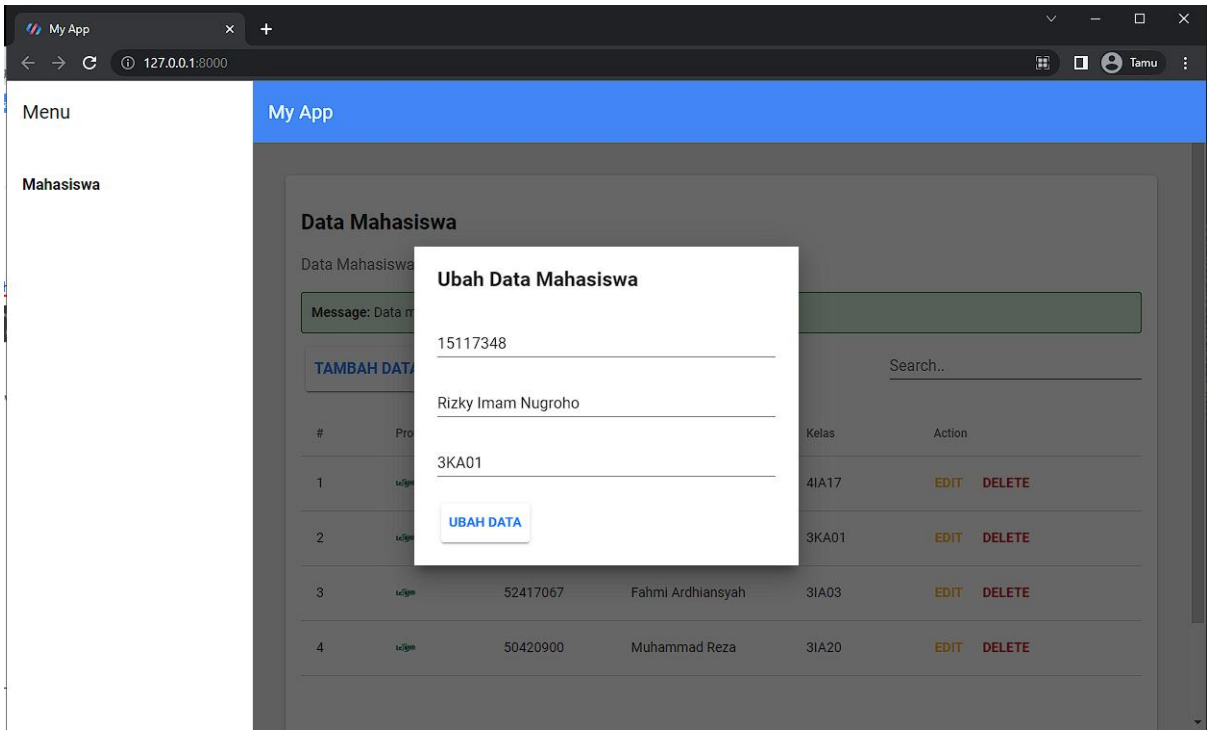
Input data



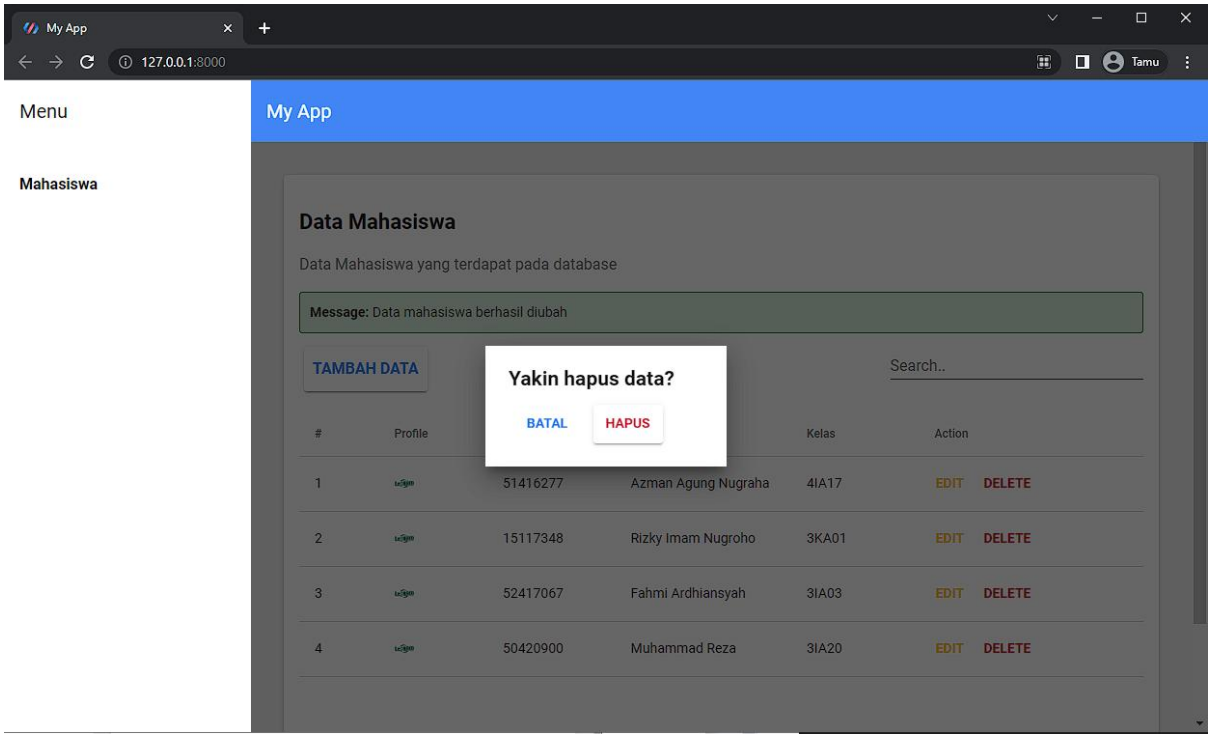
Data yang di input



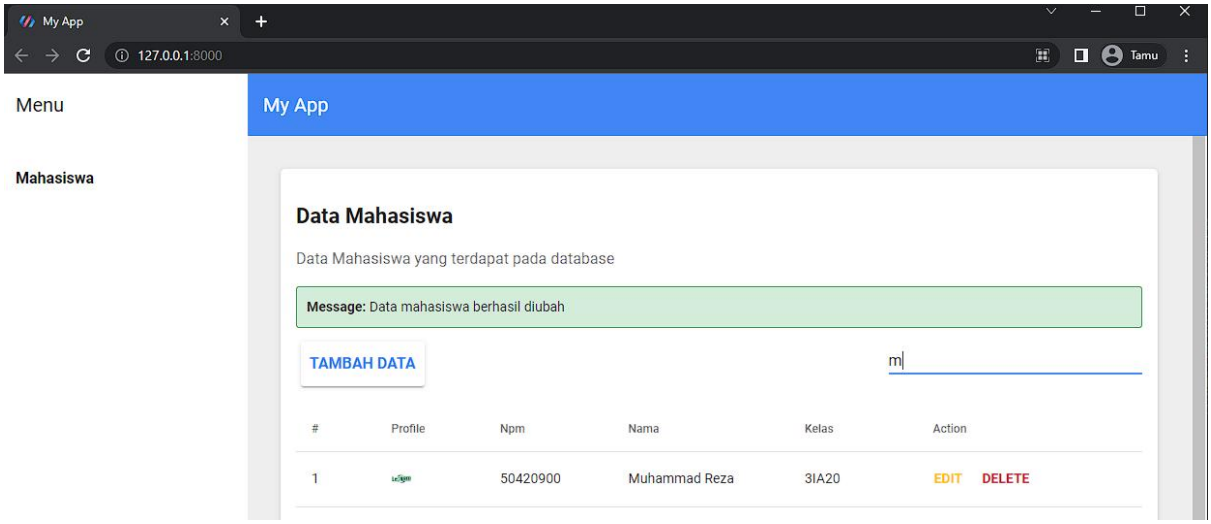
Edit data



Hapus data



Search



DATABASE

undefined - pertemuan7.sql

```
Setting environment for using XAMPP for Windows.
R@DESKTOP-Q3HDCLA c:\xampp
# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.4.25-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE reza_50420900_pert7;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> use reza_50420900_pert7;
Database changed
MariaDB [reza_50420900_pert7]> CREATE TABLE mahasiswa (
    ->     id int(6) unsigned AUTO_INCREMENT primary key,
    ->     npm char(8) NOT NULL,
    ->     nama varchar(30) NOT NULL,
    ->     kelas char(5) NOT NULL,
    ->     profile varchar(30)NOT NULL);
Query OK, 0 rows affected (0.018 sec)

MariaDB [reza_50420900_pert7]> desc mahasiswa;
+---------+-----------------+------+-----+---------+----------------+
| Field   | Type            | Null | Key | Default | Extra          |
+---------+-----------------+------+-----+---------+----------------+
| id      | int(6) unsigned | NO   | PRI | NULL    | auto_increment |
| npm     | char(8)         | NO   |     | NULL    |                |
| nama    | varchar(30)     | NO   |     | NULL    |                |
| kelas   | char(5)         | NO   |     | NULL    |                |
| profile | varchar(30)     | NO   |     | NULL    |                |
+---------+-----------------+------+-----+---------+----------------+
5 rows in set (0.007 sec)

MariaDB [reza_50420900_pert7]>
```

undefined - pertemuan7.sql

```
Setting environment for using XAMPP for Windows.
R@DESKTOP-Q3HDCLA c:\xampp
# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 61
Server version: 10.4.25-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use reza_50420900_pert7;
Database changed
MariaDB [reza_50420900_pert7]> select * from mahasiswa;
+----+----------+---------------------+-------+-------------+
| id | npm      | nama                | kelas | profile     |
+----+----------+---------------------+-------+-------------+
|  1 | 51416277 | Azman Agung Nugraha | 4IA17 | gambar1.jpg |
|  3 | 52417067 | Fahmi Ardhiansyah   | 3IA03 | gambar1.jpg |
|  4 | 50420900 | Muhammad Reza       | 3IA20 | gambar1.jpg |
+----+----------+---------------------+-------+-------------+
3 rows in set (0.000 sec)

MariaDB [reza_50420900_pert7]>
```