

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 dalam suatu database atau table. Fungsi read kurang lebih memiliki fungsi yang sama dengan perintah select ataupun search.
- ◆ Update => merupakan fungsi yang sering digunakan untuk melakukan perubahan pada data dalam suatu database. Perintah update adalah perintah yang digunakan untuk melakukan modifikasi pada data yang sudah disimpan 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
1 func createMahasiswa(w http.ResponseWriter, r *http.Request) {
2     w.Header().Set("Content-type", "application/json")
3
4     var response ResponseMessage
5
6     db := dbConn()
7     defer db.Close()
8
9     err := r.ParseForm()
10    if err != nil {
11        log.Print(err.Error())
12    }
13
14    npm := r.Form.Get("npm")
15    nama := r.Form.Get("nama")
16    kelas := r.Form.Get("kelas")
17    profile := "gambar1.jpg"
18
19    rows, err := db.Prepare("INSERT INTO mahasiswa(npm, nama, kelas, profile) VALUES(?, ?, ?, ?)")
20
21    if err != nil {
22        log.Print(err.Error())
23    }
24
25    rows.Exec(npm, nama, kelas, profile)
26
27    response.Status = true
28    response.Message = "Mahasiswa berhasil ditambahkan"
29
30    log.Print(response.Message)
31
32    json.NewEncoder(w).Encode(response)
33 }
```

Struktur Update

```
reza_50420900_pert7 - main.go

1 func updateMahasiswa(w http.ResponseWriter, r *http.Request) {
2     w.Header().Set("Content-type", "application/json")
3
4     var response ResponseMessage
5     var responseErr ResponseError
6     var mahasiswa Mahasiswa
7
8     db := dbConn()
9     defer db.Close()
10
11     err := r.ParseForm()
12     if err != nil {
13         log.Print(err.Error())
14     }
15
16     id := r.Form.Get("id")
17     npm := r.Form.Get("npm")
18     nama := r.Form.Get("nama")
19     kelas := r.Form.Get("kelas")
20
21     rows := db.QueryRow("SELECT id FROM mahasiswa WHERE id=?", id)
22     if err := rows.Scan(&mahasiswa.Id); err != nil && err == sql.ErrNoRows {
23         responseErr.Status = false
24         responseErr.Error = "Mahasiswa tidak ditemukan"
25         w.WriteHeader(http.StatusNotFound)
26         json.NewEncoder(w).Encode(responseErr)
27         return
28     }
29
30     update, err := db.Prepare("UPDATE mahasiswa SET npm=?, nama=?, kelas=? WHERE id=?")
31
32     if err != nil {
33         log.Print(err.Error())
34     }
35
36     update.Exec(npm, nama, kelas, id)
37     response.Status = true
38     response.Message = "Data mahasiswa berhasil diubah"
39
40     log.Print(response.Message)
41
42     json.NewEncoder(w).Encode(response)
43     return
44 }
```

Struktur Delete

```
reza_50420900_pert7 - main.go
1 func deleteMahasiswa(w http.ResponseWriter, r *http.Request) {
2     w.Header().Set("Content-type", "application/json")
3
4     var mahasiswa Mahasiswa
5     var response ResponseMessage
6     var responseErr ResponseError
7
8     db := dbConn()
9     defer db.Close()
10
11     params := mux.Vars(r)
12
13     rows := db.QueryRow("SELECT id FROM mahasiswa WHERE id=?", params["id"])
14     if err := rows.Scan(&mahasiswa.Id); err != nil && err == sql.ErrNoRows {
15         responseErr.Status = false
16         responseErr.Error = "Mahasiswa tidak ditemukan"
17         w.WriteHeader(http.StatusNotFound)
18         json.NewEncoder(w).Encode(responseErr)
19         return
20     }
21
22     delete, err := db.Prepare("DELETE FROM mahasiswa WHERE id=?")
23
24     if err != nil {
25         log.Print(err.Error())
26     }
27
28     delete.Exec(params["id"])
29
30     response.Status = true
31     response.Message = "Data mahasiswa berhasil dihapus"
32
33     log.Print(response.Message)
34
35     json.NewEncoder(w).Encode(response)
36     return
37 }
```

```
reza_50420900_pert7 - main.go
1 func main() {
2     r := mux.NewRouter().StrictSlash(true)
3     mime.AddExtensionType(".js", "application/javascript")
4
5     r.HandleFunc("/api/mahasiswa", getAllMahasiswa).Methods("GET")
6     r.HandleFunc("/api/mahasiswa/{id}", getMahasiswa).Methods("GET")
7     r.HandleFunc("/api/mahasiswa", createMahasiswa).Methods("POST")
8     r.HandleFunc("/api/mahasiswa", updateMahasiswa).Methods("PUT")
9     r.HandleFunc("/api/mahasiswa/{id}", deleteMahasiswa).Methods("DELETE")
10
11     r.HandleFunc("/mahasiswa/search/{keyword}", getMahasiswaByName).Methods("GET")
12
13     spa := spaHandler{staticPath: "polymer", indexPath: "index.html"}
14     r.PathPrefix("/").Handler(spa)
15
16     srv := &http.Server{
17         Handler:      r,
18         Addr:           "127.0.0.1:8000", // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
19         WriteTimeout:  15 * time.Second,
20         ReadTimeout:   15 * time.Second,
21     }
22
23     log.Print("Server berjalan di http://127.0.0.1:8000") // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
24     srv.ListenAndServe()
25 }
```

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

SQL COMMAND

```
undefined - pertemuan7.sql

1  --Membuat database reza_50420900_pert7
2  CREATE DATABASE reza_50420900_pert7;
3
4  --menggunakan database reza_50420900_pert7
5  use reza_50420900_pert7;
6
7  --membuat tabel mahasiswa
8  CREATE TABLE mahasiswa (
9      id int(6) unsigned AUTO_INCREMENT primary key,
10     npm char(8) NOT NULL,
11     nama varchar(30) NOT NULL,
12     kelas char(5) NOT NULL,
13     profile varchar(30)NOT NULL);
14
15 --menampilkan tabel employee
16 select * from mahasiswa;
17
18 --menampilkan database employee
19 desc mahasiswa;
```

Main.go

```
1 package main
2
3 import (
4     "database/sql"
5     "encoding/json"
6     "fmt"
7     "log"
8     "mime"
9     "net/http"
10    "os"
11    "path/filepath"
12    "time"
13
14    _ "github.com/go-sql-driver/mysql"
15    "github.com/gorilla/mux"
16)
17
18 type spotondier struct {
19     staticPath string
20     indexPath string
21 }
22
23 type Mahasiswa struct {
24     Id      int    `json:"id"`
25     Npm     string `json:"npm"`
26     Nama    string `json:"nama"`
27     Kelas   string `json:"kelas"`
28     Profile string `json:"profile"`
29 }
30
31 type ResponseAllData struct {
32     Status bool    `json:"status"`
33     Data   []Mahasiswa `json:"data"`
34 }
35
36 type ResponseData struct {
37     Status bool    `json:"status"`
38     Data   Mahasiswa `json:"data"`
39 }
40
41 type ResponseMessage struct {
42     Status bool    `json:"status"`
43     Message string `json:"message"`
44 }
45
46 type ResponseError struct {
47     Status bool    `json:"status"`
48     Error  string `json:"error"`
49 }
50
51 func dbConn() (db *sql.DB) {
52     dbDriver := "mysql"
53     dbName := "rezs56420980_port7" // Diganti menjadi nama database kalian masing-masing
54     dbUser := "root"
55     dbPass := ""
56     db, err := sql.Open(dbDriver, dbUser+"@"+dbPass+"@tcp(localhost)/"+dbName)
57
58     if err != nil {
59         panic(err.Error())
60     }
61
62     return db
63 }
64
65 func getAllMahasiswa(w http.ResponseWriter, r *http.Request) {
66     w.Header().Set("Content-type", "application/json")
67
68     var response ResponseAllData
69     var mahasiswa Mahasiswa
70     var mhs []Mahasiswa
71
72     db := dbConn()
73     rows, err := db.Query("SELECT * FROM mahasiswa")
74     defer db.Close()
75
76     if err != nil {
77         log.Println(err.Error())
78     }
79
80     for rows.Next() {
81         err := rows.Scan(&mahasiswa.Id, &mahasiswa.Npm, &mahasiswa>Nama, &mahasiswa.Kelas, &mahasiswa.Profile)
82
83         if err != nil {
84             log.Println(err.Error())
85         } else {
86             mhs = append(mhs, mahasiswa)
87         }
88     }
89
90     response.Status = true
91     response.Data = mhs
92
93     json.NewEncoder(w).Encode(response)
94     return
95 }
96
97 func getMahasiswa(w http.ResponseWriter, r *http.Request) {
98     w.Header().Set("Content-type", "application/json")
99
100    var response ResponseData
101    var responseErr ResponseError
102    var mahasiswa Mahasiswa
103
104    db := dbConn()
105    defer db.Close()
106
107    params := mux.Vars(r)
108
109    rows := db.QueryRow("SELECT * FROM mahasiswa WHERE id=?", params["id"])
110    err := rows.Scan(&mahasiswa.Id, &mahasiswa.Npm, &mahasiswa>Nama, &mahasiswa.Kelas, &mahasiswa.Profile)
111
112    if err != nil && err == sql.ErrNoRows {
113        responseErr.Status = false
114        responseErr.Error = "Mahasiswa tidak ditemukan"
115        w.WriteHeader(http.StatusNotFound)
116        json.NewEncoder(w).Encode(responseErr)
117        return
118    }
119
120    response.Status = true
121    response.Data = mahasiswa
122
123    json.NewEncoder(w).Encode(response)
124    return
125 }
126
127 func getMahasiswaByName(w http.ResponseWriter, r *http.Request) {
128     w.Header().Set("Content-type", "application/json")
129
130     var mahasiswa Mahasiswa
131     var mhs []Mahasiswa
132     var response ResponseAllData
133
134     db := dbConn()
135     defer db.Close()
136
137     params := mux.Vars(r)
138     query := fmt.Sprintf("SELECT * FROM mahasiswa WHERE nama LIKE '%s'", params["keyword"])
139
140     rows, err := db.Query(query)
141
142     if err != nil {
143         log.Println(err.Error())
144     }
145
146     for rows.Next() {
147         if err := rows.Scan(&mahasiswa.Id, &mahasiswa.Npm, &mahasiswa>Nama, &mahasiswa.Kelas, &mahasiswa.Profile); err != nil {
148             log.Println(err.Error())
149         }
150
151         mhs = append(mhs, mahasiswa)
152     }
153
154     response.Status = true
155     response.Data = mhs
156
157     json.NewEncoder(w).Encode(response)
158     return
159 }
```

```

mux_50420900_port7_mango

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:
            "127.0.0.1:8080", // 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://127.0.0.1:8080") // 2 angka belakang port diganti menjadi 2 angka dibelakang npm masing-masing
    srv.ListenAndServe()
}

```


Mahasiswa.html

[illegible]

B. OUPUT PROGRAM

Go run main.go

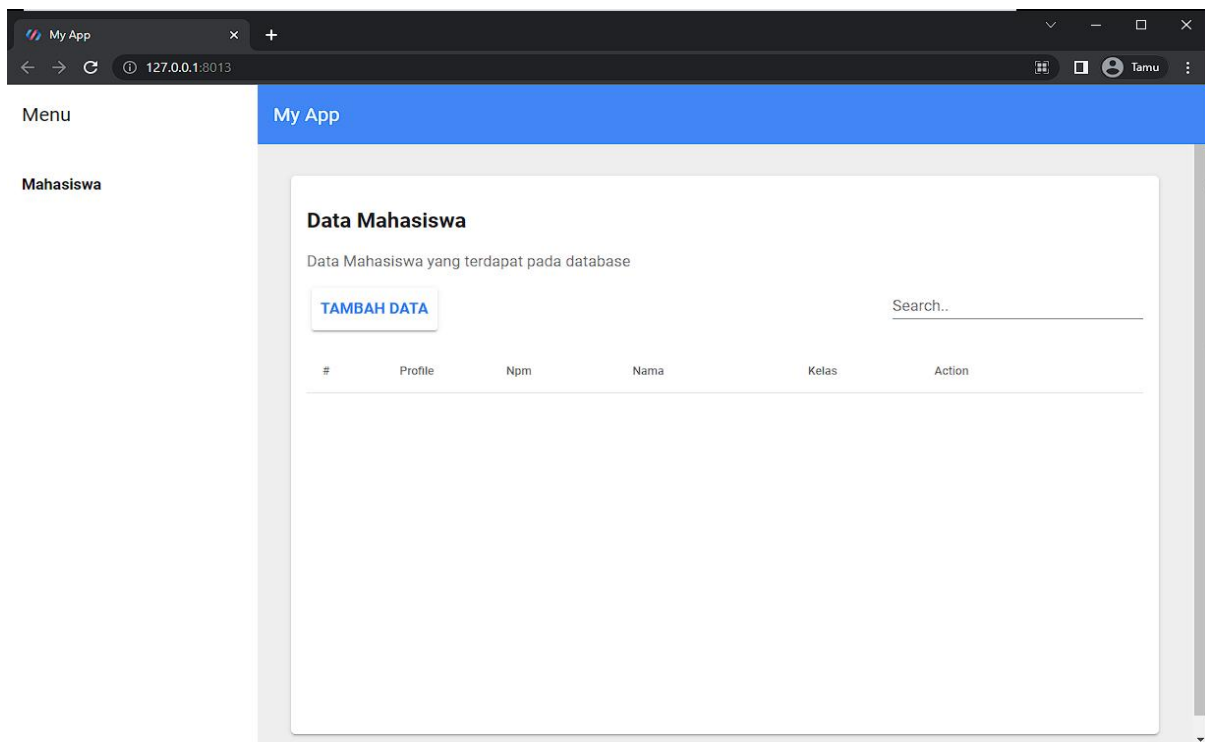
```
reza_50420900_pert7 - main.go

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

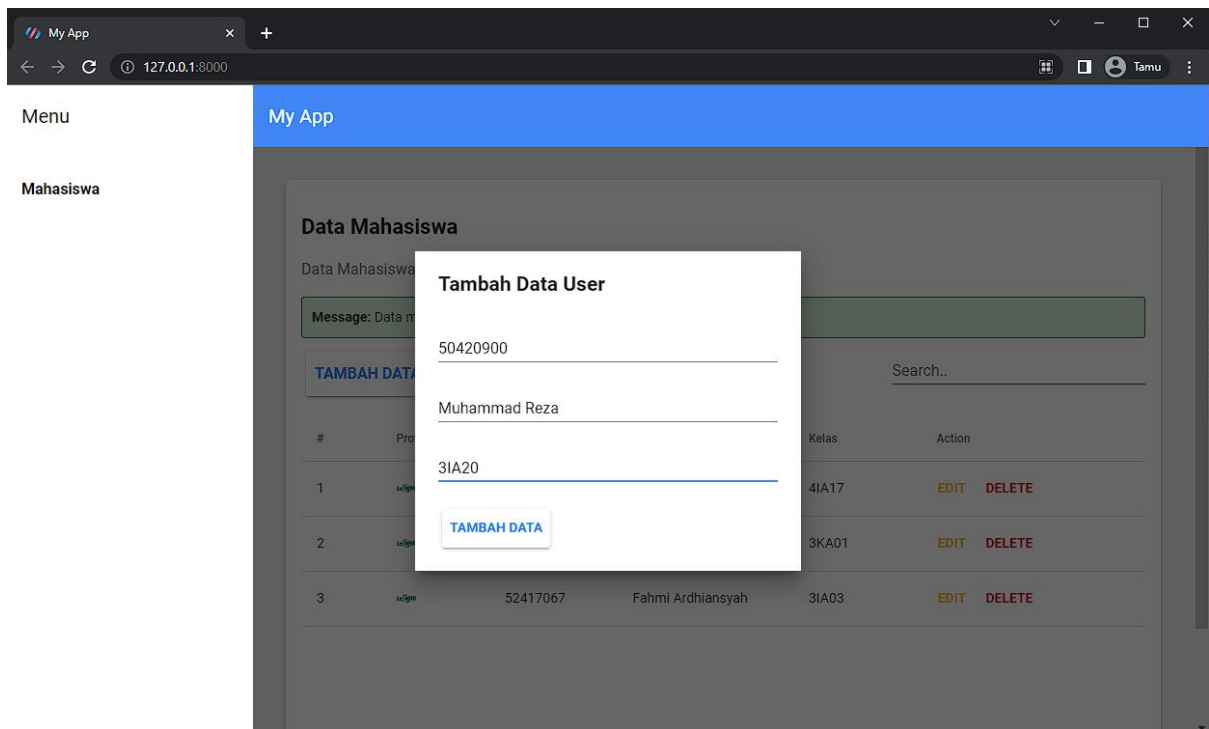
Try the new cross-platform PowerShell https://aka.ms/pscore6

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
```

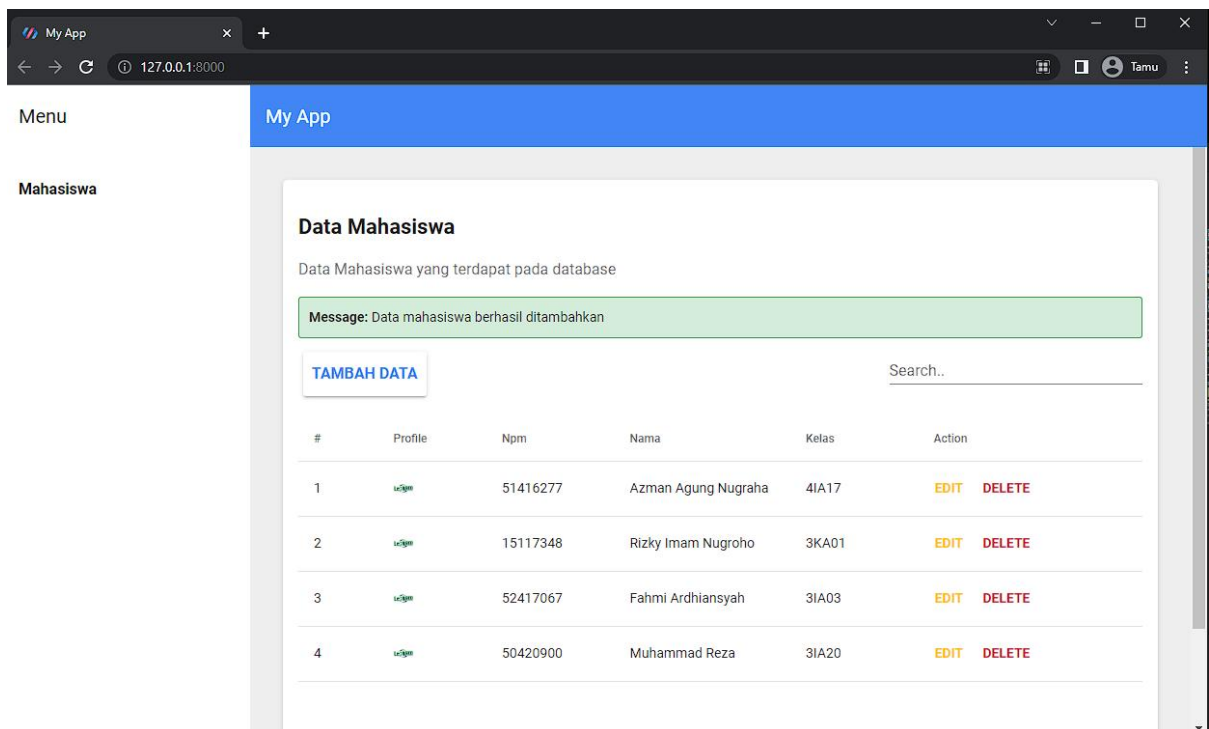
<http://127.0.0.1:8000/>



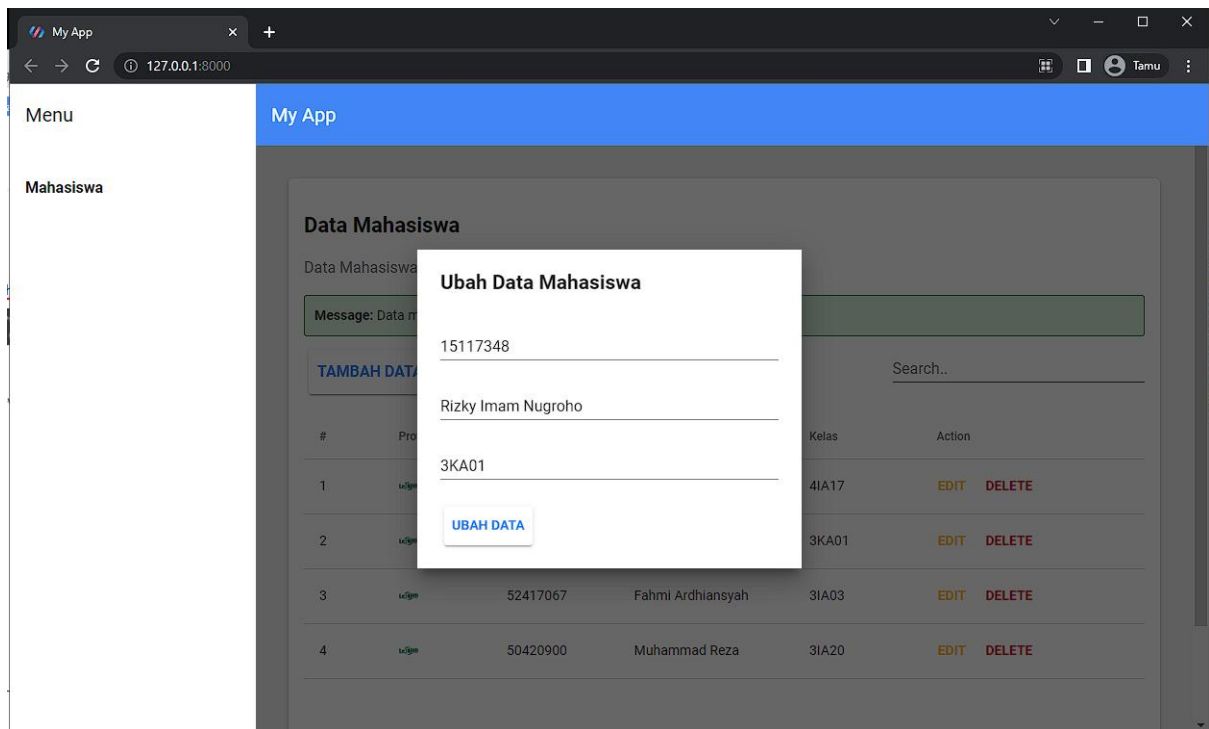
Input data



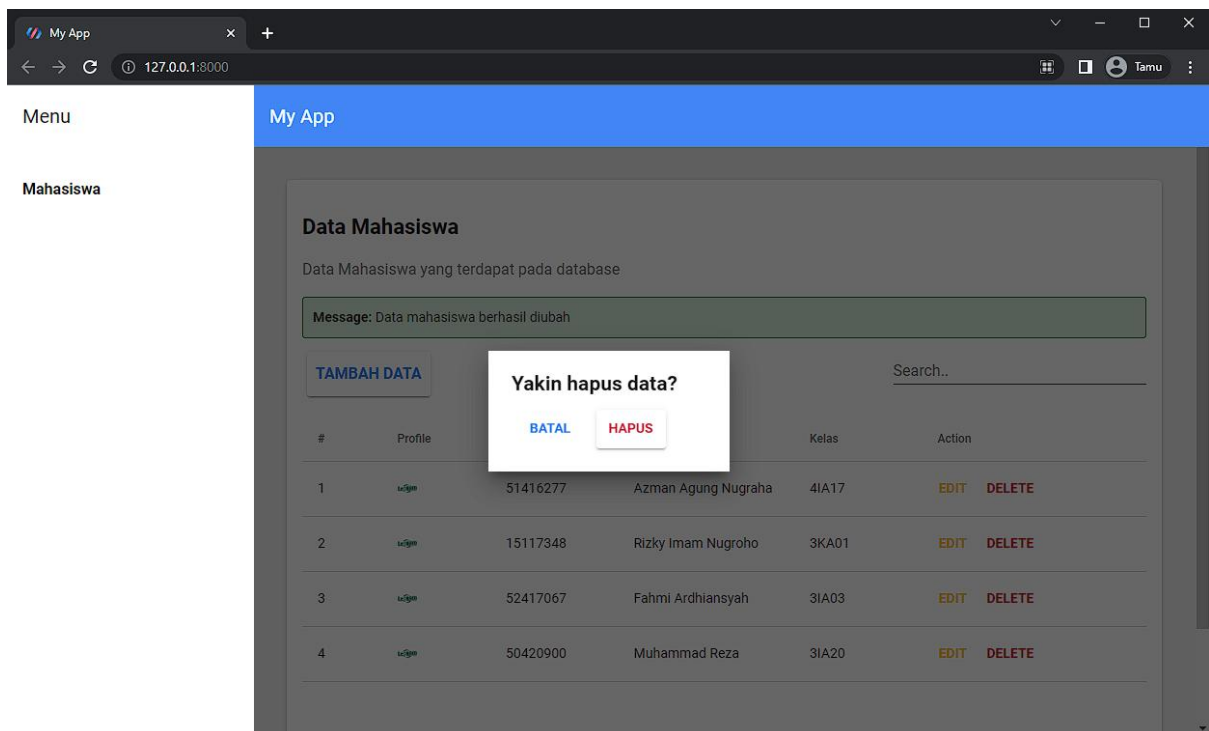
Data yang di input



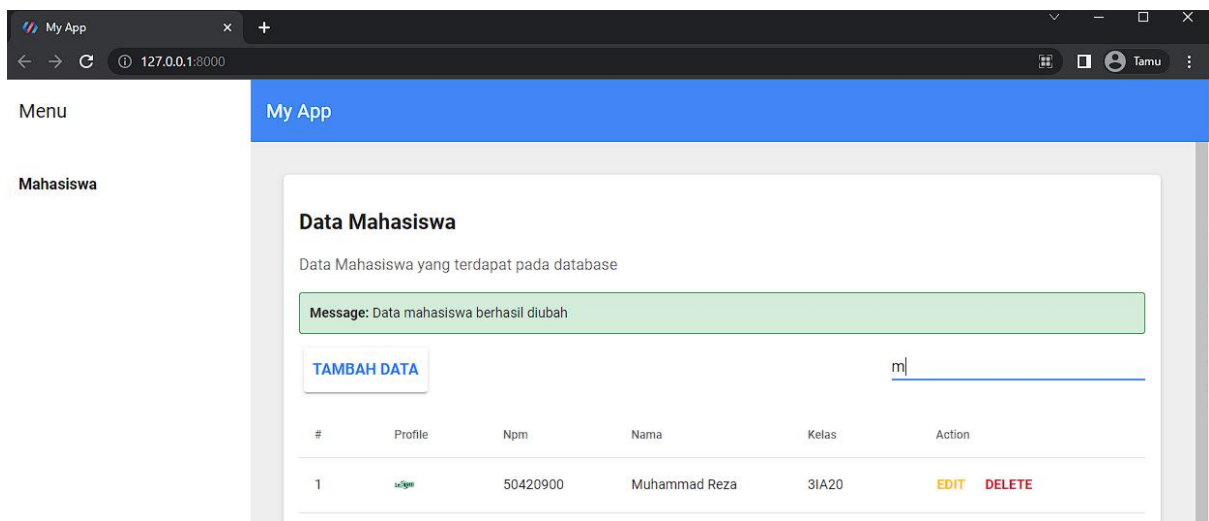
Edit data



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]>