LAPORAN PRAKTIKUM

TEKNOLOGI CLOUD COMPUTING

PERTEMUAN KE – 06

Disusun Oleh :

NIM : 195610007

Nama : Ara Widhi Astutik

Kelas : Sistem Informasi-1

UNIVERSITAS TEKNOLOGI DIGITAL INDONESIA

2021/2022

BAB I

DATA AS A SERVICE

PEMBAHASAN

Latihan

1. Install Go, MySQL, dan MongoDB
   a. Installasi Go

```
C:\Users\Asus>D:

D:\>go version
go version go1.18 windows/amd64
```

Uraian :

Pada langkah ini merupakan langkah installasi Go, namun sebelumnya saya sudah pernah menginstallnya maka saya hanya akan mengecek versi dari Go yang saya gunakan.

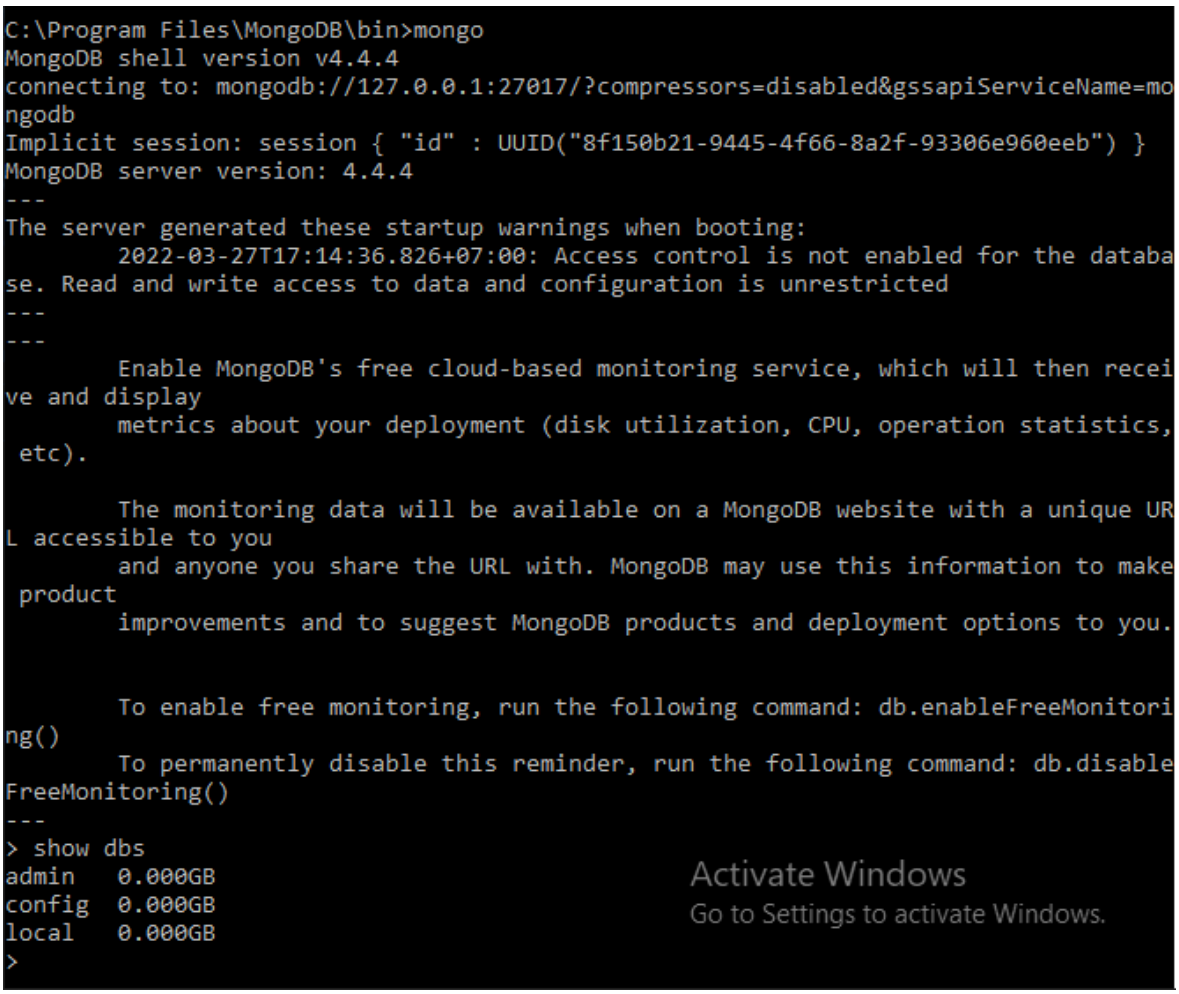
   b. Installasi MySQL

```
c:\xampp\mysql\bin>mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.26 MySQL Community Server (GPL)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| akademik           |
| amelia203          |
| cdcol              |
| dbkampus           |
| kampus             |
| kampus2            |
| klinik             |
| mysql              |
| northwind          |
```

Uraian :

Begitupun juga dengan mysql, pada tampilan output diatas menampilkan beberapa database MySQL yang pernah saya buat sebelumnya. Penggunaan mysql -u root –p digunakan untuk proses login pada MySQL.

c. Installasi MongoDB



Uraian :

Tampilan output diatas menampilkan daftar database yang telah dibuat, namun karena dalam hal ini saya belum pernah membuat database menggunakan Mongodb maka hanya akan menampilkan 3 database yang tersimpan yaitu Admin, Config, dan Local.

2. Buat 2 contoh program Go masing-masing untuk koneksi dan membaca data dari MySQL dan MongoDB.
   a. Program Go membaca data dari MySQL
      Pada langkah program Go ini nantinya akan menampilkan data, menambah data, mengubah data, menghapus data, dan mengkoneksikan ke database MySql

      - tampil.go

```
<!DOCTYPE html>
<html>
<head>
    <title>Tampil Mahasiswa - UTDI</title>
</head>
<body>
  <h2>Data Mahasiswa</h2>
  <p>{[.Pesan}}</p>
  <a href="/?aksi=tambah"><button>Tambah</button></a>
  <table border="1" cellpadding="3" cellspacing="0">
    <thead>
      <tr>
        <th>NIM</th>
        <th>NAMA</th>
        <th>PROGRAM STUDI</th>
        <th>SEMESTER</th>
        <th>AKSI</th>
      </tr>
    </thead>
    <tbody>
      {{ range $key, $value : = .Data}}
      <tr>
        <td>{{$value.NIM}}</td>
        <td>{{$value.NAMA}}</td>
        <td>{{$value.PRODI}}</td>
        <td>{{$value.SMT}}</td>
        <td>
          <a href="/?aksi=ubah&nim={{$value.NIM}}">Ubah</a>
```

```
            <a href="/?aksi=hapus&nim={{$value.NIM}}">Hapus</a>
          </td>
        </tr>
      </tbody>
    </table>
</body>
</html>
```

Uraian :

Pada tampilan ini nantinya akan dibuat pengisian data diri mahasiswa dimana terdapat nim,nama,prodi,semester. Nantinya pada bagian ini disampingnya terdapat pilihan untuk menambahkan , mengubah (mengedit) dan menghapus data.

- tambah.go

```
<!DOCTYPE html>
<html>
<head>
  <title>Tambah Mahasiswa - UTDI</title>
</head>
<body>
  <h2>TAMBAH MAHASISWA</h2>
  <a href="/">Batal</a>
  <hr>
  <form method="POST" action="/tambah">
    <label>NIM : <br>
      <input type="text" name="nim" value="">
    </label>
    <label>NAMA : <br>
      <input type="text" name="nama" value="">
    </label>
    <label>PROGRAM STUDI : <br>
      <input type="text" name="prodi" value="">
    </label>
    <label>SEMESTER : <br>
      <input type="text" name="smt" value="">
```

```
      </label>
      <br>
      <input type="submit" value="Tambah"/>
   </form>
</body>
</html>
```

Uraian :

Pada tampilan menu tambah, user dapat memasukkan kembali data diri mahasiswa tersebut, juga dapat membatalkannya apabila tidak ingin menambahkan data. Menggunakan perintah submit untuk melakukan proses tambah. Fungsinya adalah untuk memproses data yang diinput.

- ubah.go

```
<!DOCTYPE html>
<html>
<head>
   <title>Ubah Mahasiswa - UTDI</title>
</head>
<body>
   <h2>UBAH MAHASISWA</h2>
   <a href="/">Batal</a>
   <hr>
   <form method="POST" action="/ubah">
      <label>NIM : <br>
         <input type="text" name="nim" value="{{ (index .Data
0).NIM}}"readonly="readonly">
      </label>
      <label>NAMA : <br>
         <input type="text" name="nama"value="{{ (index .Data 0).NAMA}}">
      </label>
      <label>PROGRAM STUDI : <br>
         <input type="text" name="progdi" value="{{ (index .Data
0).PROGDI}}">
      </label>
      <label>SEMESTER : <br>
```

```
        <input type="number" name="smt" value="{{ (index .Data 0).SMT}}">
      </label>
      <br>
      <input type="submit" value="Ubah"/>
    </form>
</body>
</html>
```

Uraian :

Untuk menu ubah, sebenarnya tidak jauh berbeda dengan menu tambah sebelumnya, namun yang membedakannya disini adalah, pada menu ubah (mengedit) menggunakan perintah index data dan atribut readonly. Atribut readonly digunakan untuk menentukan bahwa elemen <input> hanya dapat dibaca (read only) tanpa bisa diedit.

- hapus.go

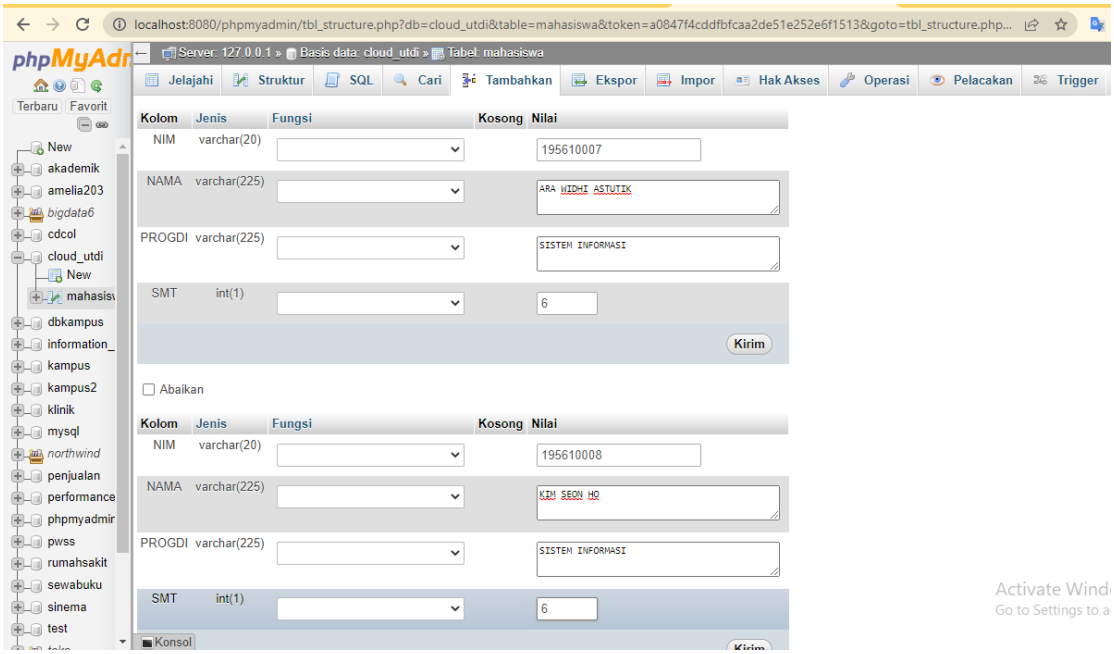
```
<!DOCTYPE html>
<html>
<head>
   <title>Ubah Mahasiswa - UTDI/title>
</head>
<body>
   <h2>HAPUS MAHASISWA</h2>
   <a href="/">Batal</a>
   <hr>
   <form method="POST" action="/hapus">
      <label>NIM : <br>
         <input type="text" name="nim" value="{{ (index .Data
0).NIM}}"readonly="readonly">
      </label>
      <label>NAMA : <br>
         <input type="text" name="nama" value="{{ (index .Data
0).NAMA}}"readonly="readonly">
      </label>
      <label>PROGRAM STUDI : <br>
```

```
        <input type="text" name="progdi" value="{{ (index .Data
0).PROGDI}}"readonly="readonly">
      </label>
      <label>SEMESTER : <br>
        <input type="number" name="smt" value="{{ (index .Data 0).SMT}}">
      </label>
      <br>
      <input type="submit" value="Hapus"/>
    </form>
</body>
</html>
```

Uraian :

Langkah untuk menu hapus sama halnya dengan menu ubah (mengedit) sebelumnya.

- mengkoneksikan ke database MySQL



Uraian :

Pada tahap koneksi ke database MySQL ini, kita terlebih dahulu membuat database tersebut pada PhpMyAdmin, dengan cara localhost kemudian membuat database baru sesuai keinginan , setelah itu mulai menentukan tabel yang akan dibuat dan tidak lupa untuk menginputkan datanya. Nantinya database ini akan dipanggil melalui Go , pada saat dipanggil nanti ia akan

menampilkan data-data yang ada dalam database sebelumnya yang telah kita buat dan isi.

- CRUD MySQL

```
import(
    "tmt"
    "net/http"
    "html/template"
    "database/sql"
    "github.com/go-sql-driver/mysql"
)
type mahasiswa struct(
    NIM string
    NAMA string
    Progdi string
    SMT int
)

type response struct(
    Status bool
    Pesan string
    Data []mahasiswa
)

func koneksi() (*sql.DB,error){
    db,salahe := sql.Open("msql","root:@tcp(127.0.0.1:3306")/cloud_utdi)
    if salahe != nil {
        return nil, salahe
    }
    return db, nil
}

func tampil(pesane string) response{
    db, salahe := koneksi()
    if salahe != nil {
```

```
    return response{
        Status : false,
        Pesan : "Gagal Koneksi: "+salahe.Error(),
        Data : []mahasiswa{},
    }
}
defer dataMhs.Close()

        var hasil []mahasiswa

for dataMhs.Next (){
    var mhs = mahasiswa{}
    var salahe = dataMhs.Scan(mhs.NIM, mhs.NAMA, mhs.Progdi, mhs.SMT)

    if salahe != nil {
        return response{
            Status : false,
            Pesan : "Gagal Baca : "+salahe.Error(),
            Data : []mahasiswa{},
        }
    }
    hasil = append(hasil, mhs)
}
salahe = dataMhs.Error();

if salahe != nil{
    return response{
        Status : false,
        Pesan : "Kesalahan : "+salahe.Error(),
        Data : []mahasiswa{},
    }
}
return response{
    Status: true,
```

```
    Pesan : pesane,
    Data : hasil,
  }
  }
  func getMhs(nim string) response{
    db, salahe := koneksi()
    if salahe != nil{
      return response{
                                Status : false,
        Pesan : "Gagal Koneksi : "+salahe.Error(),
        Data : []mahasiswa{},
      }
  }
  defer db.close()

  dataMhs, salahe := db.Query("select * from mahasiswa where nim=?",nim)
  if salahe != nil{
    return response{
    Status : false,
    Pesan : "Gagal Query : "+salahe.Error(),
    Data : []mahasiswa{},
    }
  }
  defer dataMhs.close()
  var hasil []mahasiswa
  for dataMhs.Next(){
    var mhs = mahasiswa{}
    var salahe = dataMhs.Scan(mhs.NIM, mhs.NAMA, mhs.PROGDI,
mhs.SMT)

  if salahe != nil {
    return response{
    Status : false,
    Pesan : "Gagal Baca  : "+salahe.Error(),
```

```
        Data : []mahasiswa{},
        }
    }
    hasil append(hasil, mhs)
    }
    salahe = dataMhs.Err()
    if salahe != nil {
        return response{
        Status : false,
        Pesan : "Kesalahan  :"+salahe.Error(),
        Data : []mahasiswa{},
        }
    }
    return response{
        Status:true,
        Pesan:"Berhasil tampil",
        Data:hasil,
        }
    }
    func tambah (nim string, nama string, progdi string, smt string)response{
        db, salahe := koneksi ()
        if salahe != nil{
            return response{
                Status : fale,
                Pesan : "Gagal Koneksi : "+salahe.Error(),
                Data : []mahasiswa{},
            }
        }
        defer db.close()
        db, salahe := db.Exec("insert into mahasiswa values
(?,?,?,?)",NIM,NAMA,PROGDI,SMT)
        if salahe != nil{
            return response{
                Status : false,
```

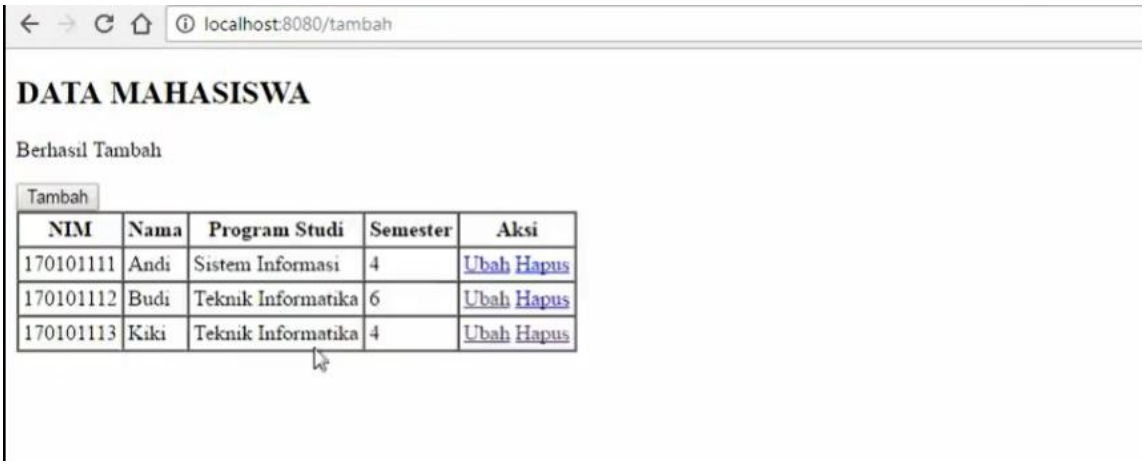
```
            Pesan : "Gagal Query insert : "+salahe.Error(),
            Data : []mahasiswa{},
        }
    }

                func ubah(nim string, nama string,progdi string, smt
string)response{
      db, salahe := koneksi ()
       if salahe != nil{
          return response{
             Status : false,
             Pesan : "Gagal Koneksi : "+salahe.Error(),
             Data : []mahasiswa{},
    }
  }
  defer db.close()
  db, salahe := db.Exec("update mahasiswa set nama -7, progdi-7, smt-7
where nim=?",nama,progdi,smt,nim)
    if salahe != nilP{
      return response{
         Status : false,
         Pesan : "Gagal Query Update : "+salahe.Error(),
         Data : []mahasiswa{},
      }
    }
    return response{
      Status : false,
      Pesan : "Gagal Koneksi : "+salahe.Error(),
      Data : []mahasiswa{},
      }
    }
    defer db.close()
    db, salahe := db.Exec("delete from  mahasiswa where nim=?",nim)
    if salahe != nil{
```

```
        return response{
            Status : false,
            Pesan : "Gagal Query Update : "+salahe.Error(),
            Data : []mahasiswa{},
        }
                }
    return respon{
        Status : true,
        Pesan : "Berhasil Hapus"
        Data : []mahasiswa{},
    }
    }
```

Uraian :

CRUD dalam hal ini digunakan untuk menerapkan pada sistem basis data yang membantu proses pengolahan informasi secara sistematis. Operasi pemograman yang diterapkan dalam hal ini yaitu membuat, membaca, mengupdate, dan menghapus data (Create, Read, Update, Delete.

- Output




localhost:8080/?aksi=ubah&nim=170101113
UBAH MAHASISWA
Batal
NIM :
170101113
Nama :
Tofik
Program Studi :
Sistem Informasi
Semester :
4
Ubah

BAB II

DAFTAR PUSTAKA

https://www.youtube.com/watch?v=ryS2hwFW73o