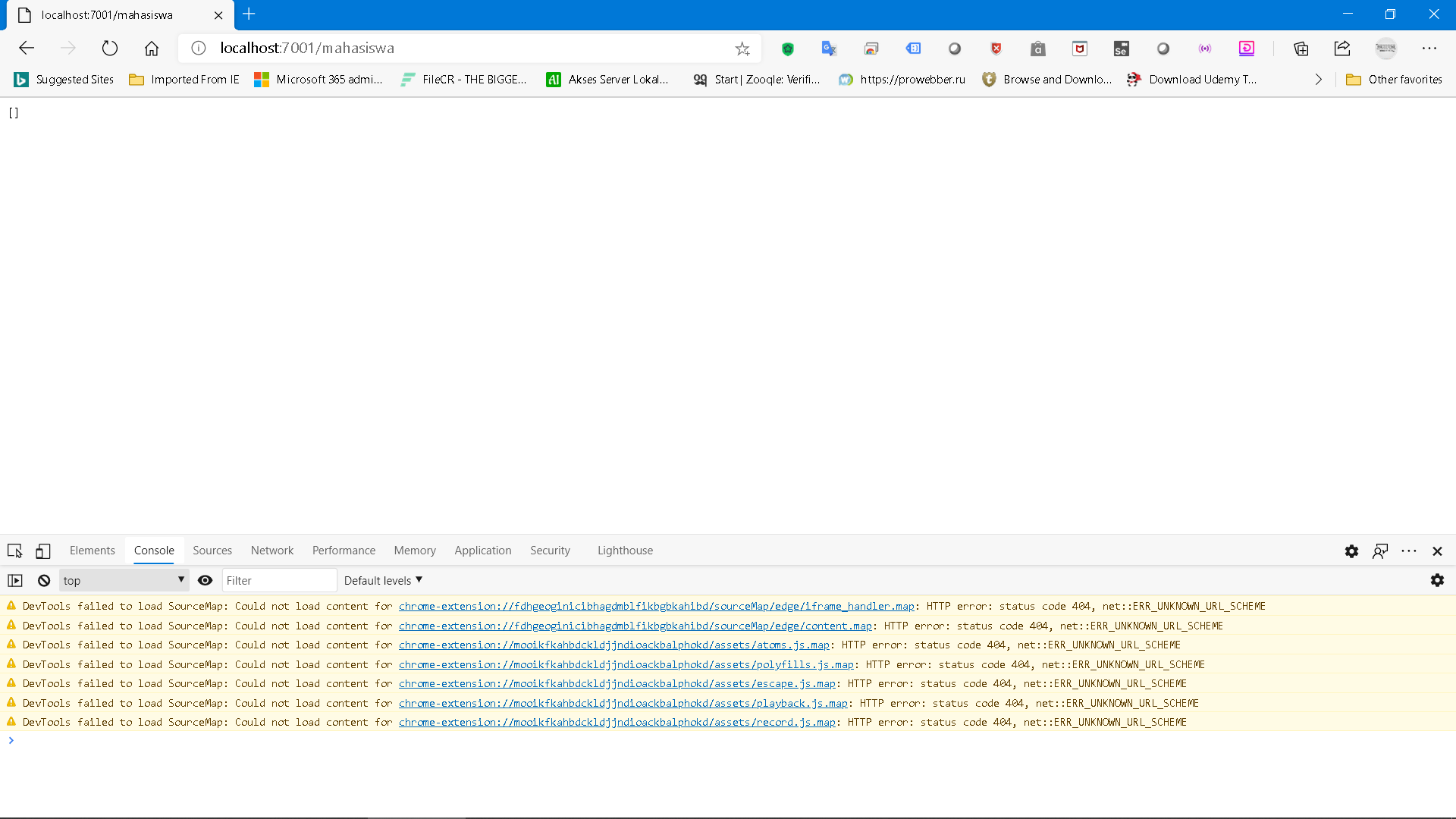
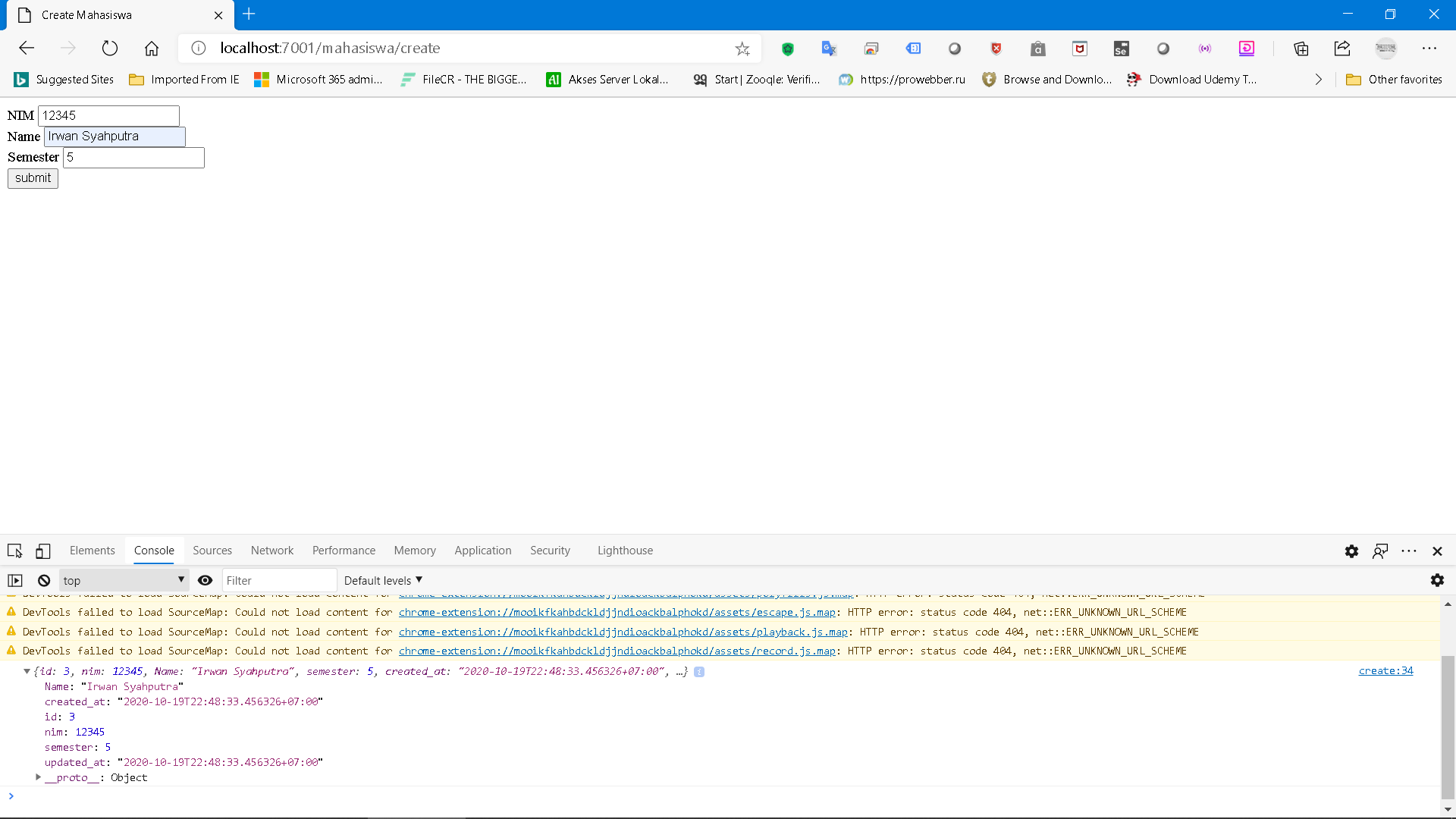
**UI-BE-12 - IT Perbankan Kelas Back End**

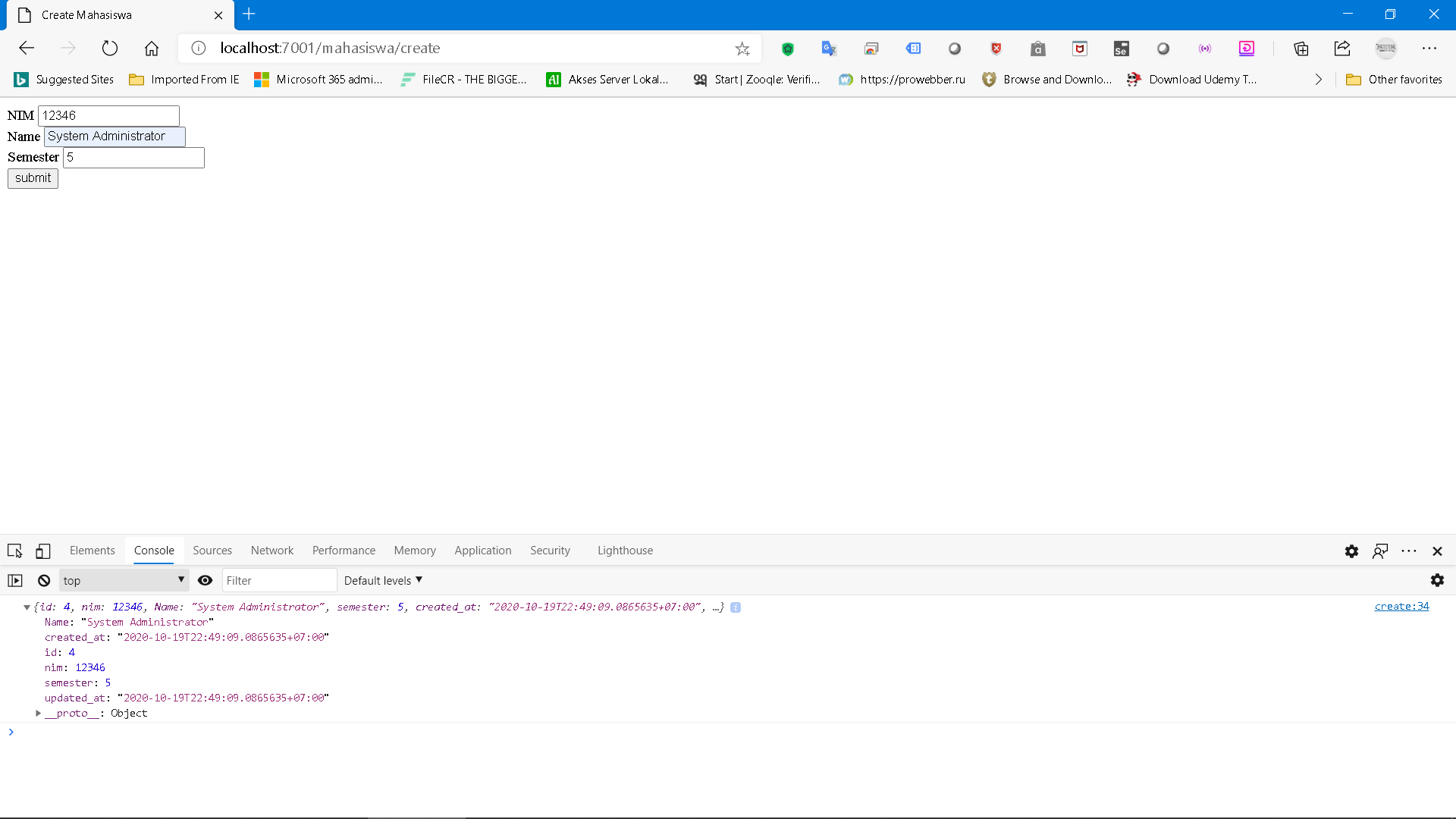
**0568281021-9 Irwan Syahputra**



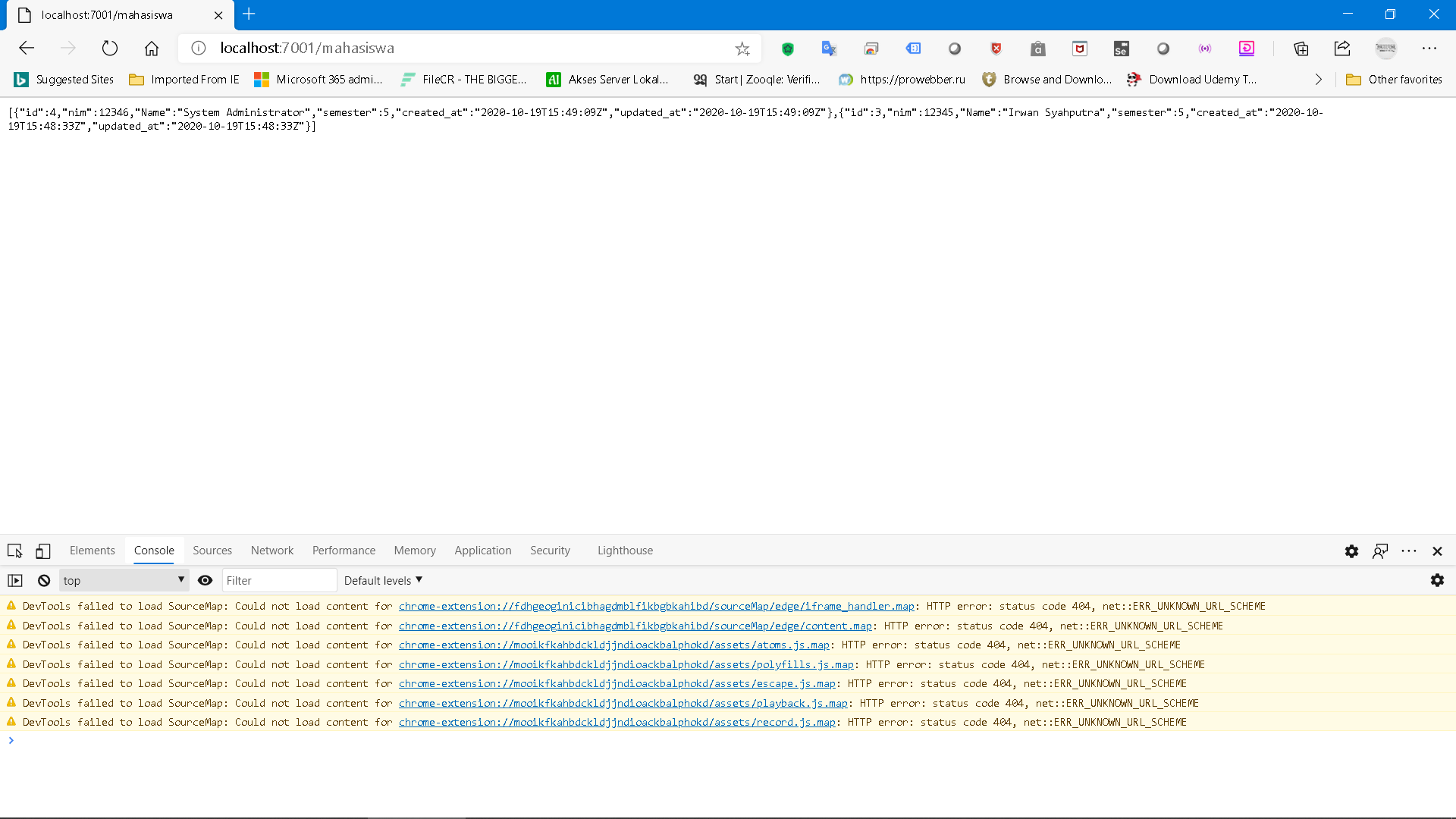
Show all data mahasiswa



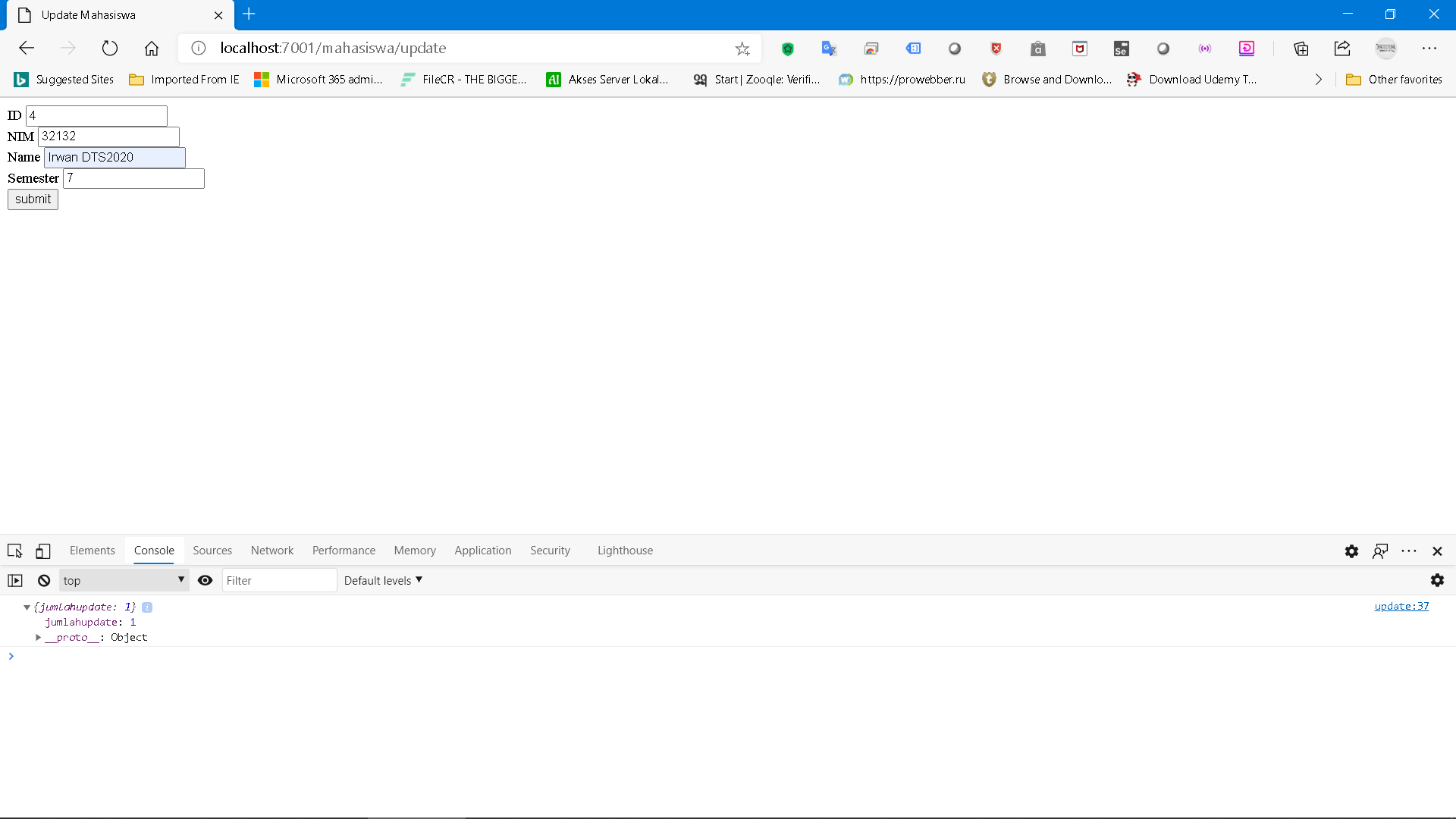
Create new mahasiswa 1



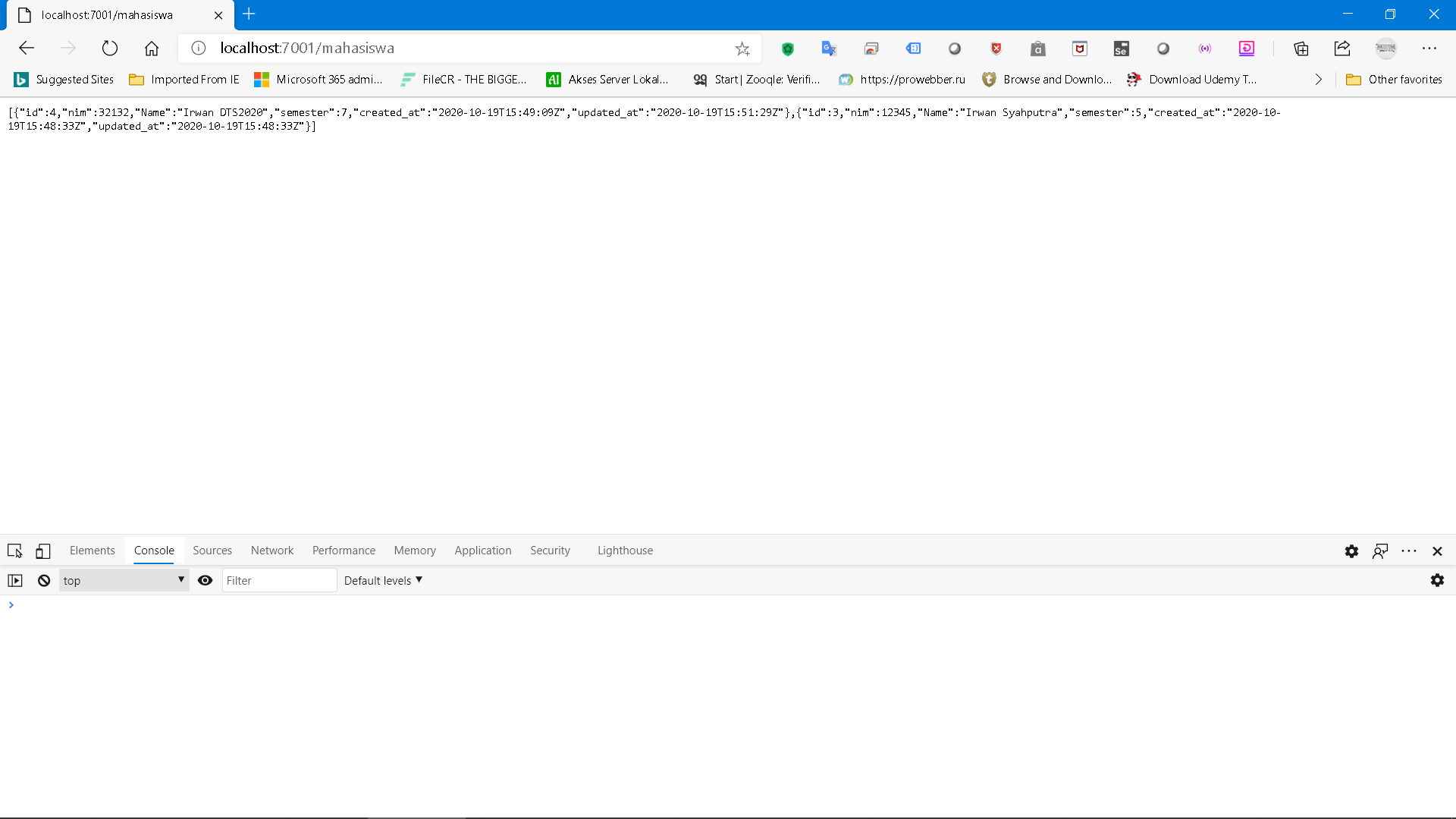
Create new mahasiswa 2



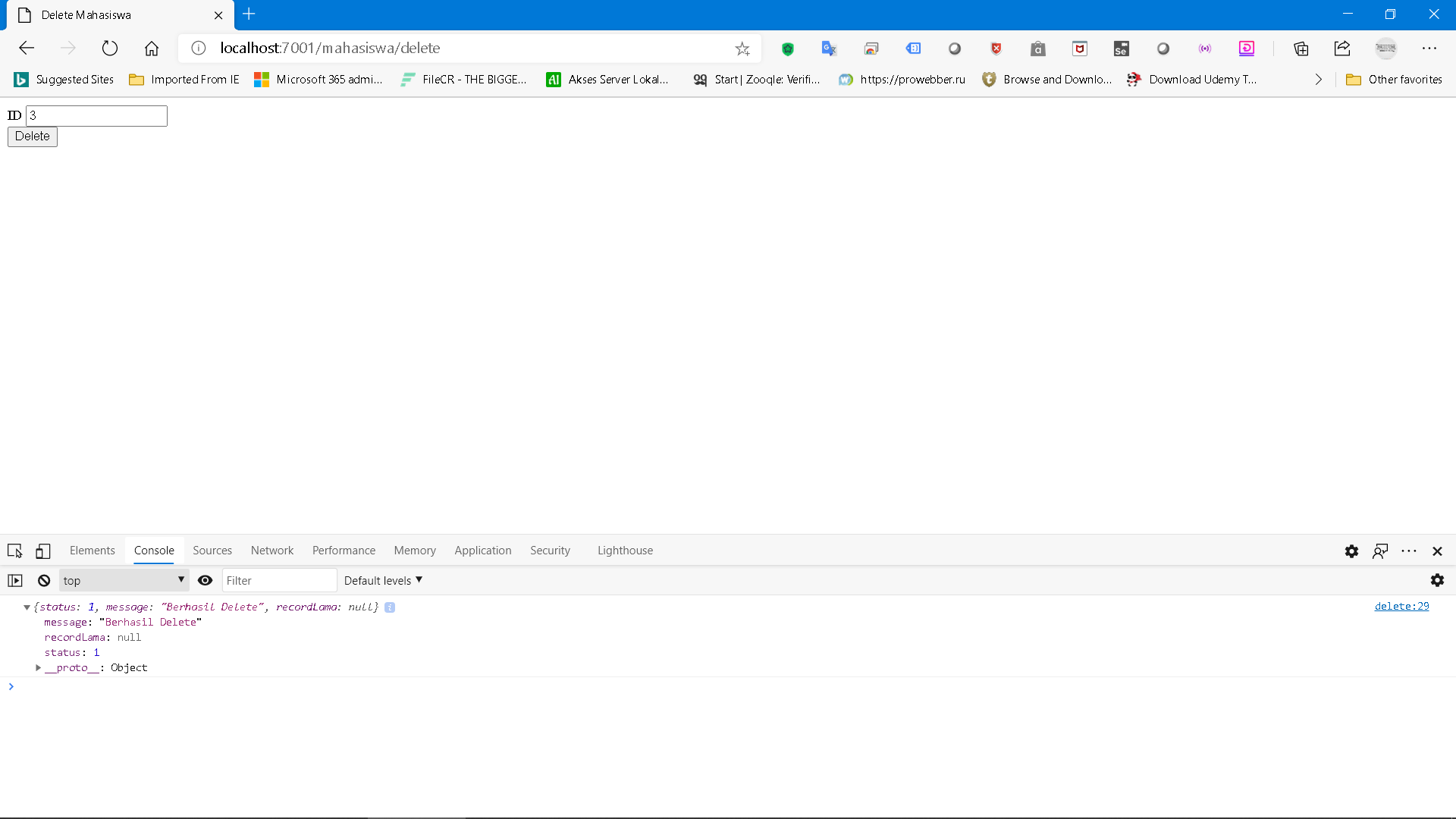
Show data mahasiswa after create



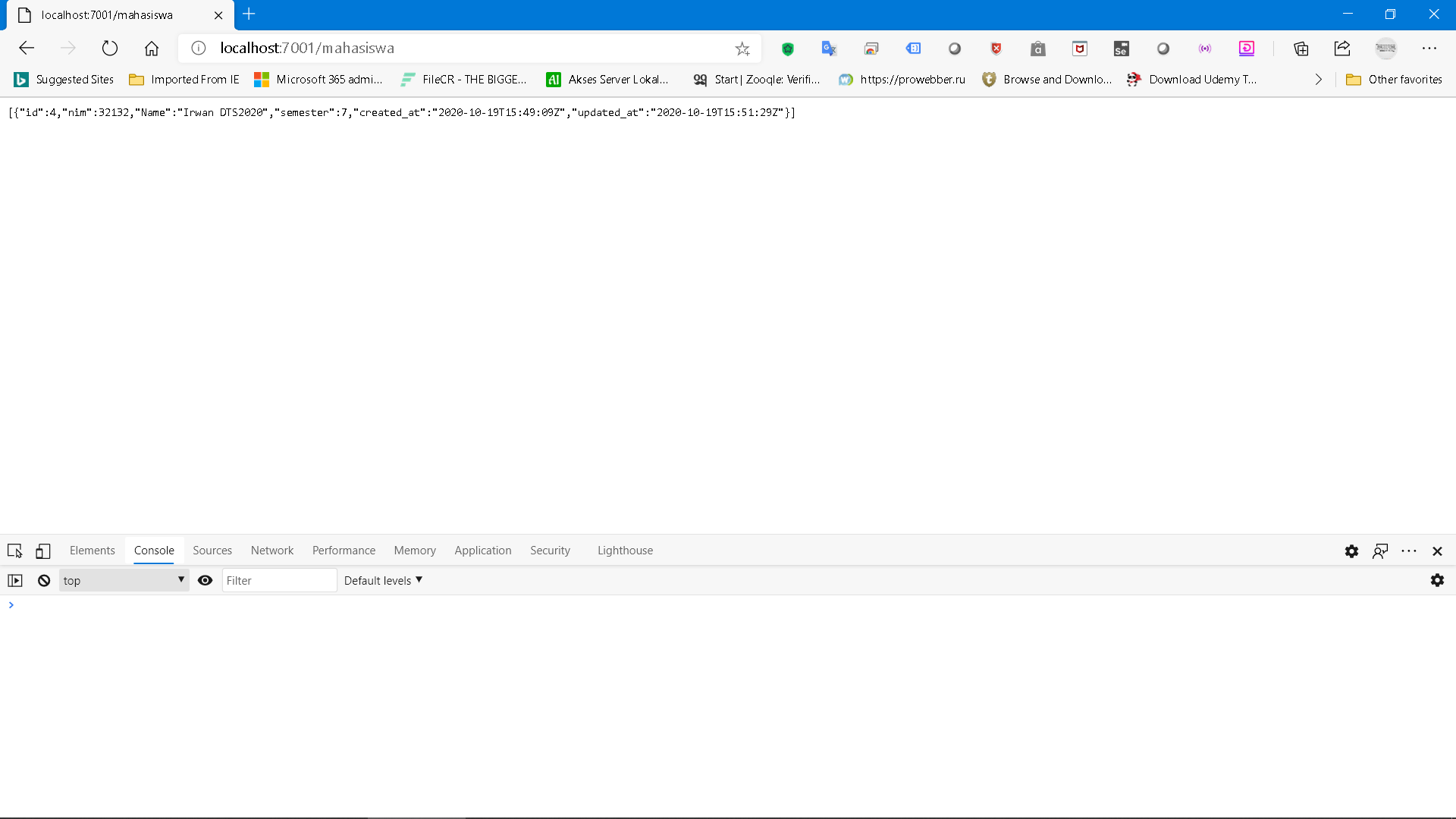
Update Mahasiswa Id 4



Show data mahasiswa after update



Delete Mahasiswa Id 3



Show After Delete

Sources Main.go

package main

import (

    "be3gomy/config"

    "be3gomy/mahasiswa"

    "be3gomy/model"

    "be3gomy/utils"

    "database/sql"

    "encoding/json"

    "fmt"

    "log"

    "net/http"

)

const addr = `:7001`

type Server struct {

    db      \*sql.DB

    ViewDir *string*

}

type handler func(w http.ResponseWriter, r \*http.Request)

type H map[*string*]interface{}

func InitServer() \*Server {

    db, err := config.Mysql()

    if err != nil {

        log.Fatal(err)

    }

    return &Server{

        db:      db,

        ViewDir: `views/`,

    }

}

func (s \*Server) Listen() {

    log.Println(`listen at ` + addr)

    http.HandleFunc(`/mahasiswa`, s.Mahasiswa())

*//http.HandleFunc(`/mahasiswa/show`, s.MahasiswaGetFromID())*

    http.HandleFunc(`/mahasiswa/create`, s.MahasiswaCreate())

    http.HandleFunc(`/mahasiswa/update`, s.MahasiswaUpdate())

    http.HandleFunc(`/mahasiswa/delete`, s.MahasiswaDelete())

    err := http.ListenAndServe(addr, nil)

    if err != nil {

        fmt.Println(err)

    }

}

func (s \*Server) Mahasiswa() handler {

    return func(w http.ResponseWriter, r \*http.Request) {

        mahasiswas, err := mahasiswa.SelectAll(s.db)

        if utils.IsError(w, err) {

            return

        }

        utils.ResponseJson(w, mahasiswas)

    }

}

func (s \*Server) MahasiswaCreate() func(http.ResponseWriter, \*http.Request) {

    return func(w http.ResponseWriter, r \*http.Request) {

        if r.Method == `GET` {

            http.ServeFile(w, r, s.ViewDir+`mahasiswa\_create.html`)

            return

        }

        m := model.Mahasiswa{}

        err := json.NewDecoder(r.Body).Decode(&m)

        if utils.IsError(w, err) {

            return

        }

        err = mahasiswa.Insert(s.db, &m)

        if utils.IsError(w, err) {

            return

        }

        utils.ResponseJson(w, m)

    }

}

func (s \*Server) MahasiswaUpdate() func(http.ResponseWriter, \*http.Request) {

    return func(w http.ResponseWriter, r \*http.Request) {

        if r.Method == `GET` {

            http.ServeFile(w, r, s.ViewDir+`mahasiswa\_update.html`)

            return

        }

        m := model.Mahasiswa{}

        ru := model.ResponseUpdate{}

        err := json.NewDecoder(r.Body).Decode(&m)

        if utils.IsError(w, err) {

            return

        }

        err = mahasiswa.Update(s.db, &m, &ru)

        if utils.IsError(w, err) {

            return

        }

        utils.ResponseJson(w, ru)

    }

}

func (s \*Server) MahasiswaDelete() func(http.ResponseWriter, \*http.Request) {

    return func(w http.ResponseWriter, r \*http.Request) {

        if r.Method == `GET` {

            http.ServeFile(w, r, s.ViewDir+`mahasiswa\_delete.html`)

            return

        }

        m := model.Mahasiswa{}

        rd := model.ResponseDelete{}

        err := json.NewDecoder(r.Body).Decode(&m)

        if utils.IsError(w, err) {

            return

        }

        err = mahasiswa.Delete(s.db, &m, &rd)

        if utils.IsError(w, err) {

            return

        }

        utils.ResponseJson(w, rd)

    }

}

func main() {

    server := InitServer()

    server.Listen()

}

package model

type ResponseDelete struct {

    Status     *int*                  `json:"status"`

    Message    *string*               `json:"message"`

    RecordLama map[*string*]Mahasiswa `json:"recordLama"`

}

package model

type ResponseUpdate struct {

    JumlahUpdate *int* `json:"jumlahupdate"`

}

Repo

package mahasiswa

import (

    "be3gomy/model"

    "database/sql"

    "fmt"

    "time"

)

const table = `mahasiswa`

const dateformat = `2006-01-02 15:04:05`

func SelectAll(db \*sql.DB) (mahasiswas []model.Mahasiswa, err *error*) {

    sql := fmt.Sprintf(`SELECT \* FROM %s ORDER BY id DESC`, table)

    rows, err := db.Query(sql)

    if err != nil {

        return nil, err

    }

    defer rows.Close()

    mahasiswas = []model.Mahasiswa{}

    for rows.Next() {

        m := model.Mahasiswa{}

        createdAt, updatedAt := ``, ``

        err = rows.Scan(&m.ID,

            &m.NIM,

            &m.Name,

            &m.Semester,

            &createdAt,

            &updatedAt)

        m.CreatedAt, \_ = time.Parse(dateformat, createdAt)

        m.UpdatedAt, \_ = time.Parse(dateformat, updatedAt)

        if err != nil {

            return nil, err

        }

        mahasiswas = append(mahasiswas, m)

    }

    return

}

func Insert(db \*sql.DB, m \*model.Mahasiswa) (err *error*) {

    sql := fmt.Sprintf(`INSERT INTO %v (nim, name, semester, created\_at, updated\_at)

VALUES(?,?,?,?,?)`, table)

    now := time.Now()

    res, err := db.Exec(sql, m.NIM, m.Name, m.Semester, now, now)

    if err != nil {

        return err

    }

    lastId, err := res.LastInsertId()

    if err != nil {

        return err

    }

    m.ID = int(lastId)

    m.CreatedAt = now

    m.UpdatedAt = now

    return

}

func Update(db \*sql.DB, m \*model.Mahasiswa, ru \*model.ResponseUpdate) (err *error*) {

    sql := fmt.Sprintf(`UPDATE %v set nim = ?, name =?, semester = ?, updated\_at = ? where id = ?`, table)

    now := time.Now()

    res, err := db.Exec(sql, m.NIM, m.Name, m.Semester, now, m.ID)

    if err != nil {

        return err

    }

    aff, err := res.RowsAffected()

    fmt.Println(aff)

    if err != nil {

        return err

    }

    ru.JumlahUpdate = int(aff)

    return

}

func Delete(db \*sql.DB, m \*model.Mahasiswa, rd \*model.ResponseDelete) (err *error*) {

    sql := fmt.Sprintf("DELETE FROM %v where id = ?", table)

    res, err := db.Exec(sql, m.ID)

    if err != nil {

        return err

    }

    aff, err := res.RowsAffected()

    fmt.Println(aff)

    if err != nil {

        return err

    }

    rd.Status = int(aff)

    rd.Message = "Berhasil Delete"

    fmt.Println(m)

    return

}

**UI-BE-12 - IT Perbankan Kelas Back End**

**0568281021-9 Irwan Syahputra**