// Import library yang digunakan

const express = require('express');

const cors = require('cors');

const bodyParser = require('body-parser');

const mysql = require('mysql')

const app = express();

// Tentukan Port yang digunakan ketika running

const port = 2019;

// Buat database connection

const conn = mysql.createConnection({

host: 'localhost',

user: 'Nur',

password: '1234abcd',

database: 'moviebertasbih',

port: 3306

});

app.use(cors());

app.use(bodyParser.json())

// Tampilan menu awal ketika localhost dijalankan

app.get('/', (req,res) => {

res.send('<center><h1> Ini HomePage !</h1></center/>')

})

// Memulai Crud Data

//========== Manage Movie ============

// Membuat table movie di mysql

//CREATE TABLE IF NOT EXISTS `movies` (

// `id` INT NOT NULL AUTO\_INCREMENT,

// `nama` VARCHAR(45) NOT NULL,

// `tahun` INT NOT NULL,

// `description` VARCHAR(45) NOT NULL,

// PRIMARY KEY (`id`));

// Lalu inser data ke table movies

// insert into movies values

// (null,'Equalizer','2012','keren abis'),

// (null,'Aquaman', '2018', ' keren coy'),

// (null,'Transformer', '2010', 'bolehlah lah');

//Menampilkan semua data movie melalui method get dari

// select semua kolom di table movies

// di postmant get => localhost:2019/movielist

app.get('/movielist', (req,res) => {

var sql = `select \* from movies`

conn.query(sql, (err,results) => {

if(err) throw err;

res.send(results);

})

})

//Menambahkan data pada table movie melalui method .post dengan menggunakan

// req.body

// insert semua data dari req.body ke dalam tabel movies

app.post('/addmovie', (req,res) => {

var newMovie = {

nama: req.body.nama,

tahun: req.body.tahun,

description: req.body.description

}

var sql = 'insert into movies set ?;'

conn.query(sql, newMovie, (err, result) => {

if (err) throw err;

console.log(result);

})

sql = `select \* from movies`

conn.query(sql, (err,result1) => {

if(err) throw err;

console.log(result1)

res.send(result1);

})

})

//Mengedit data pada table movie melalui method .put dengan menggunakan

// params

// select data yang mau diedit lalu diisi melalui req.body

// lalu update tabel movie melalui params.id

//dipostman digunakan : localhost:2019/editmovie/id (id yang diselect)

// lalu di req.body diisi data yang mo diedit

app.put('/editmovie/:id', (req, res) => {

var editMovie = {

nama: req.body.nama,

tahun: req.body.tahun,

description: req.body.description

}

var movieId = req.params.id;

var sql = `Update movies set ? where id = ${movieId}`;

conn.query(sql,editMovie, (err,result) => {

if(err) throw err;

// res.send(results)

sql = `select \* from movies;`

conn.query(sql, (err,results1) => {

if(err) throw err;

res.send(results1)

})

})

})

//Menentukan id dari data pada tabel movies yang mo di delete melalui

// req.params.id

// dipostman digunakan : localhost:2019/deletemovie/id (id yang diselect)

app.delete('/deletemovie/:id', (req,res)=> {

var movieId = req.params.id;

var sql = `DELETE from movies where id = ${movieId};`

conn.query(sql, (err,results) => {

if(err) throw err

sql = `DELETE from movcat where idMovie =${movieId};`

conn.query(sql,(err1,results1) => {

if(err1) throw err1

res.send(results1);

})

})

}

)

//========== Manage Category ============

// Membuat table categories di mysql

// CREATE TABLE IF NOT EXISTS `categories` (

// `id` INT NOT NULL AUTO\_INCREMENT,

// `nama` VARCHAR(45) NOT NULL,

// PRIMARY KEY (`id`));

// Lalu inser data ke table movies

// insert into categories values

// (null,'Action'),

// (null,'Drama'),

// (null,'Comedy'),

// (null,'Fantasy'),

// (null,'Horor');

//Menampilkan semua data categories melalui method get dari

// select semua kolom di table categories

// di postmant get => localhost:2019/categorylist

app.get('/categorylist', (req,res) => {

var sql = `select \* from categories`

conn.query(sql, (err,results) => {

if(err) throw err;

res.send(results);

})

})

//Menambahkan data pada table categories melalui method .post dengan menggunakan

// req.body

// insert semua data dari req.body ke dalam tabel categories

//dipostman digunakan post : localhost:2019/addcategory

// lalu di req.body diisi data yang mo ditambahkan

app.post('/addcategory', (req,res) => {

var newCat = {

nama: req.body.nama

}

var sql = 'insert into categories set ?;'

conn.query(sql, newCat, (err, result) => {

if (err) throw err;

console.log(result);

})

sql = `select \* from categories`

conn.query(sql, (err1,result1) => {

if(err1) throw err1;

console.log(result1)

res.send(result1);

})

})

//Mengedit data pada table categories melalui method .put dengan menggunakan

// params

// select data yang mau diedit lalu diisi melalui req.body

// lalu update tabel categories melalui params.id

//dipostman digunakan : localhost:2019/editcategory/id (id yang diselect)

// lalu di req.body diisi data yang mo diedit

app.put('/editcategory/:id', (req, res) => {

var editCat = {

nama: req.body.nama,

}

var catId = req.params.id;

var sql = `Update categories set ? where id = ${catId}`;

conn.query(sql,editCat, (err,result) => {

if (err) throw err

sql = `select \* from categories;`

conn.query(sql, (err,results1) => {

if(err) throw err;

res.send(results1)

})

})

})

//Menentukan id dari data pada tabel categories yang mo di delete melalui

// req.params.id

// dipostman digunakan : localhost:2019/deletecategory/id (id yang diselect)

app.delete('/deletecategory/:id', (req,res)=> {

var catId = req.params.id;

sql = `DELETE from categories where id = ${catId};`

conn.query(sql, (err,results) => {

if(err) throw err

sql = `DELETE from movcat where idCategory = ${catId};`

conn.query(sql,(err,results1) => {

if(err) throw err

res.send(results1);

})

})

}

)

//========== Connect Movies and Category ============

// Membuat table movcat di mysql

//CREATE TABLE IF NOT EXISTS `movcat` (

// `idmovie` INT NOT NULL,

// `idcategory` INT NOT NULL);

// Lalu insert data ke table movcat

// insert into movcat values

// (1, 1), (2,1),(2,2),(1,2),(3,1),(3,3);

//Menampilkan semua data movcat melalui method get dari

// select dari querynya (var sql dengan menggunakan join)

// di postmant get => localhost:2019/connectionlist

app.get('/connectionlist', (req,res) => {

var sql = `select m.id as idMovie, m.nama as namaMovie,

c.id as idCategory, c.nama as namaCategory

from movies m

join movcat mc

on m.id = mc.idMovie

join categories c

on mc.idCategory = c.id;`

conn.query(sql, (err,results) => {

if(err) throw err;

res.send(results);

})

})

//Menambahkan data pada table movcat melalui method .post dengan menggunakan

// req.body

// insert semua data dari req.body ke dalam tabel movcat

//dipostman digunakan post : localhost:2019/addconnection

// lalu di req.body diisi data yang mo ditambahkan (idMovie, idCategory)

app.post('/addconnection', (req,res) => {

var {idMovie, idCategory} = req.body

var sql = `insert into movcat value (${idMovie},${idCategory})`

conn.query(sql, (err,results) => {

if(err) throw err

res.send(results)

console.log(results)

})

})

//Menentukan data pada tabel movcat yang mo di delete melalui

// req.body

// dipostman digunakan : localhost:2019/deleteconnection

// lalu masukkan req.body (idMovie, idCategory)

app.delete('/deleteconnection', (req,res)=> {

var {idMovie, idCategory} = req.body

var sql = `delete from movcat

where idMovie =${idMovie} and idCategory =${idCategory}`

conn.query(sql, (err,results) => {

if(err) throw err

res.send(results)

console.log(results)

})

})

//untuk merunning API menggunakan method listen

app.listen(port, () => console.log('API aktif di port ' + port));