Tugas Wulan (project mandiri)

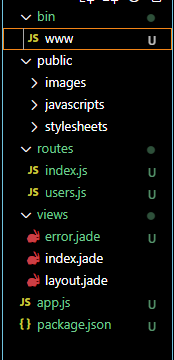
Membuat operasi CRUD NodeJs menggunakan express, mysql, dan database

Referensi materi: <https://w3path.com/node-js-crud-operation-with-express-and-mysql-database/amp/>

1. **Membuat database (nama: project), lalu buat table ‘customers’ mengisi data**

INSERT INTO `customers` (`id`, `nama`, `email`) VALUES (NULL, 'Catur', 'cwuland25@gmail.com'), (NULL, 'Wulandari', 'wulandary402057@gmail.com'), (NULL, 'Catur Wulansari', 'cwulan942@gmail.com'), (NULL, 'Arju Syafa\'ah', '18104020044@student.uin-suka.ac.id'), (NULL, 'Dillatun Nafisah', '18104020046@student.uin-suka.ac.id'), (NULL, 'Erina Kusumawati', '18104020048@student.uin-suka.ac.id'), (NULL, 'Fithratun Nafs', '18104020050@student.uin-suka.ac.id'), (NULL, 'Farhatul Ghina', '18104020052@student.uin-suka.ac.id'), (NULL, 'Ghayatul Hayah', '18104020054@student.uin-suka.ac.id'), (NULL, 'Himmatul Ulya', '18104020056@student.uin-suka.ac.id');

1. **Membuat folder expressfirst di vs code**
2. **Ketik di terminal (arahkan di folder expressfirst):**
3. npx express-generator



1. npm install -g express-generator
2. express CrudDemo --view ejs
3. cd CrudDemo (masuk ke folder CrudDemo)
4. npm install
5. cd ../ (agar berada di folder expressfirst)
6. install dependensi:

npm install express-flash –save  
 npm install express-session --save  
 npm install express-validator --save  
 npm install method-override --save  
 npm install mysql –save

1. membuat folder lib. Didalam folder lib buatlah file baru dengan nama db.js, lalu masukkan kodingan berikut:

var mysql=require('mysql');  
 var connection=mysql.createConnection({  
 host:'localhost',  
 user:'your username',  
 password:'your password',  
 database:'your database name'  
 });  
connection.connect(function(error){  
 if(!!error){  
 console.log(error);  
 }else{  
 console.log('Connected!:)');  
 }  
 });   
module.exports = connection;

1. **Buka file app.js, lalu ubah kodingan sesuai dgn berikut:**

var createError = require('http-errors');  
 var express = require('express');  
 var path = require('path');  
 var cookieParser = require('cookie-parser');  
 var logger = require('morgan');  
 var expressValidator = require('express-validator');  
 var flash = require('express-flash');  
 var session = require('express-session');  
 var bodyParser = require('body-parser');  
  
 var mysql = require('mysql');  
 var connection = require('./lib/db');  
  
 var indexRouter = require('./routes/index');  
 var usersRouter = require('./routes/users');  
 var customersRouter = require('./routes/customers');  
  
 var app = express();  
  
// view engine setup  
 app.set('views', path.join(\_\_dirname, 'views'));  
 app.set('view engine', 'ejs');  
  
 app.use(logger('dev'));  
 app.use(bodyParser.json());  
 app.use(bodyParser.urlencoded({ extended: true }));  
 app.use(cookieParser());  
 app.use(express.static(path.join(\_\_dirname, 'public')));  
  
 app.use(session({   
 secret: '123456cat',  
 resave: false,  
 saveUninitialized: true,  
 cookie: { maxAge: 60000 }  
 }))  
  
 app.use(flash());  
 app.use(expressValidator());  
  
 app.use('/', indexRouter);  
 app.use('/users', usersRouter);  
 app.use('/customers', customersRouter);  
  
 // catch 404 and forward to error handler  
 app.use(function(req, res, next) {  
 next(createError(404));  
 });  
  
 // error handler  
 app.use(function(err, req, res, next) {  
 // set locals, only providing error in development  
 res.locals.message = err.message;  
 res.locals.error = req.app.get('env') === 'development' ? err : {};  
 // render the error page  
 res.status(err.status || 500);  
 res.render('error');  
 });  
 module.exports = app;

1. **buatlah folder routes, lalu buat file didalamnya dengan nama customers.js, lalu ketik:**

var express = require('express');

var router = express.Router();

var connection = require('../lib/db');

/\* GET home page. \*/

router.get('/', function(req, res, next) {

connection.query('SELECT \* FROM customers ORDER BY id desc',function(err,rows) {

if(err){

req.flash('error', err);

res.render('customers',{page\_title:"Customers - Node.js",data:''});

}else{

res.render('customers',{page\_title:"Customers - Node.js",data:rows});

}

});

});

// SHOW ADD USER FORM

router.get('/add', function(req, res, next){

// render to views/user/add.ejs

res.render('customers/add', {

title: 'Add New Customers',

name: '',

email: ''

})

})

// ADD NEW USER POST ACTION

router.post('/add', function(req, res, next){

req.assert('name', 'Name is required').notEmpty() //Validate name

req.assert('email', 'A valid email is required').isEmail() //Validate email

var errors = req.validationErrors()

if( !errors ) { //No errors were found. Passed Validation!

var user = {

name: req.sanitize('name').escape().trim(),

email: req.sanitize('email').escape().trim()

}

connection.query('INSERT INTO customers SET ?', user, function(err, result) {

//if(err) throw err

if (err) {

req.flash('error', err)

// render to views/user/add.ejs

res.render('customers/add', {

title: 'Add New Customer',

name: user.name,

email: user.email

})

} else {

req.flash('success', 'Data added successfully!');

res.redirect('/customers');

}

})

}

else { //Display errors to user

var error\_msg = ''

errors.forEach(function(error) {

error\_msg += error.msg + '<br>'

})

req.flash('error', error\_msg)

/\*\*

\* Using req.body.name

\* because req.param('name') is deprecated

\*/

res.render('customers/add', {

title: 'Add New Customer',

name: req.body.name,

email: req.body.email

})

}

})

// SHOW EDIT USER FORM

router.get('/edit/(:id)', function(req, res, next){

connection.query('SELECT \* FROM customers WHERE id = ' + req.params.id, function(err, rows, fields) {

if(err) throw err

// if user not found

if (rows.length <= 0) {

req.flash('error', 'Customers not found with id = ' + req.params.id)

res.redirect('/customers')

}

else { // if user found

// render to views/user/edit.ejs template file

res.render('customers/edit', {

title: 'Edit Customer',

//data: rows[0],

id: rows[0].id,

name: rows[0].name,

email: rows[0].email

})

}

})

})

// EDIT USER POST ACTION

router.post('/update/:id', function(req, res, next) {

req.assert('name', 'Name is required').notEmpty() //Validate nam //Validate age

req.assert('email', 'A valid email is required').isEmail() //Validate email

var errors = req.validationErrors()

if( !errors ) {

var user = {

name: req.sanitize('name').escape().trim(),

email: req.sanitize('email').escape().trim()

}

connection.query('UPDATE customers SET ? WHERE id = ' + req.params.id, user, function(err, result) {

//if(err) throw err

if (err) {

req.flash('error', err)

// render to views/user/add.ejs

res.render('customers/edit', {

title: 'Edit Customer',

id: req.params.id,

name: req.body.name,

email: req.body.email

})

} else {

req.flash('success', 'Data updated successfully!');

res.redirect('/customers');

}

})

}

else { //Display errors to user

var error\_msg = ''

errors.forEach(function(error) {

error\_msg += error.msg + '<br>'

})

req.flash('error', error\_msg)

/\*\*

\* Using req.body.name

\* because req.param('name') is deprecated

\*/

res.render('customers/edit', {

title: 'Edit Customer',

id: req.params.id,

name: req.body.name,

email: req.body.email

})

}

})

// DELETE USER

router.get('/delete/(:id)', function(req, res, next) {

var user = { id: req.params.id }

connection.query('DELETE FROM customers WHERE id = ' + req.params.id, user, function(err, result) {

//if(err) throw err

if (err) {

req.flash('error', err)

// redirect to users list page

res.redirect('/customers')

} else {

req.flash('success', 'Customer deleted successfully! id = ' + req.params.id)

// redirect to users list page

res.redirect('/customers')

}

})

})

module.exports = router;

1. **buat folder views, dalam folder tsb buat folder customers, lalu buat tiga file: add.ejs, index.ejs, dan edit,ejs.**
2. **Setelah selesai, jalankan perintah: npm start**
3. **Lalu buka browser dan ketik:** [**http://localhost:3000/customers**](http://localhost:3000/customers)