**Лабораторна робота**

**З дисципліни:**

**Програмування WEB додатків**

**На тему:** Promise

Виконав:

Студент 3 курсу

Групи ПЗС-2044

Бойко Денис

**Лабораторна робота**

**Тема:** Promise

**Мета:** Навчитись використувати Promise.

Хід роботи

Лістинг Бекенду з використанням Promise:

const http = require("http");

const fs = require("fs");

const { Client } = require('pg')

const client = new Client({

user: 'postgres',

host: 'localhost',

database: 'mob\_phone',

password: '1234566',

port: 5432,

})

client.connect();

http.createServer(function (request, response) {

if (request.url.split('?')[0] === "/addscreen" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_screen = +params[0].split('=')[1];

const name\_screen = +params[1].split('=')[1];

let sql = "INSERT INTO screen (id\_screen , name\_screen) VALUES (" + id\_screen + ',' + name\_screen + ")";

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/delscreen" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM screen WHERE id\_screen =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/addcpu" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_cpu = +params[0].split('=')[1];

const name\_cpu = +params[1].split('=')[1];

let sql = "INSERT INTO cpu (id\_cpu , name\_cpu) VALUES (" + id\_cpu + ',' + name\_cpu + ")";

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/delcpu" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM cpu WHERE id\_cpu =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/addbrend" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_brend = +params[0].split('=')[1];

const name\_brend = +params[1].split('=')[1];

let sql = "INSERT INTO brend (id\_brend , name\_brend) VALUES (" + id\_brend + ',' + name\_brend + ")";

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/delbrend" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM brend WHERE id\_brend =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/addsklad" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_poz = +params[0].split('=')[1];

const id\_teleph = +params[1].split('=')[1];

const k\_st = +params[2].split('=')[1];

let sql = "INSERT INTO sklad (id\_poz , id\_teleph,k\_st) VALUES (" + id\_poz + ',' + id\_teleph + ',' + k\_st + ")";

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/desklad" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM sklad WHERE id\_poz =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/addoperation" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_operation = +params[0].split('=')[1];

const id\_client = +params[1].split('=')[1];

const id\_teleph = +params[2].split('=')[1];

let sql = "INSERT INTO operation (id\_operation , id\_client,id\_teleph) VALUES (" + id\_operation + ',' + id\_client + ',' + id\_teleph + ")";

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/deloperation" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM operation WHERE id\_operation =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/addclient" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_client = +params[0].split('=')[1];

const PIB\_client = +params[1].split('=')[1];

const teleph = +params[2].split('=')[1];

let sql = "INSERT INTO client (id\_client , pib\_client,teleph) VALUES (" + id\_client + ',' + PIB\_client + ',' + teleph + ")";

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/delclient" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM client WHERE id\_client =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/addteleph" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const id\_teleph = +params[0].split('=')[1];

const model\_teleph =+params[1].split('=')[1];

const id\_brend = +params[2].split('=')[1];

const id\_cpu = +params[3].split('=')[1];

const id\_screen = +params[4].split('=')[1];

let sql = 'INSERT INTO teleph (id\_teleph , model\_teleph,id\_brend,id\_cpu,id\_screen) VALUES (' + id\_teleph + ',' + model\_teleph + ',' + id\_brend + ',' + id\_cpu + ',' + id\_screen + '); select \* from teleph';

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

if (request.url.split('?')[0] === "/delteleph" && request.method === "GET") {

const params = request.url.split('?')[1].split('&')

const table\_name = +params[0].split('=')[1];

let sql = "DELETE FROM teleph WHERE id\_teleph =" + table\_name;

client

.query(sql)

.then(result => {

console.log(sql)

response.writeHead(200, { "Content-Type": "application/json" })

})

.catch(e => console.error(e.stack))

.then(() => console.log(sql))

}

///////////////////////

else if (request.url === "/brend" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM brend')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/screen" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM screen')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/cpu" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM cpu')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/sklad" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM skald')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/teleph" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM teleph')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/teleph" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM teleph')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/client" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM client')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/teleph\_view" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM teleph\_view')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

else if (request.url === "/operation" && request.method === "GET") {

response.writeHead(200, { "Content-Type": "application/json" });

let mas = [];

let row

client

.query('SELECT \* FROM operation')

.then(result => {

for (row of result.rows) { mas.push(row); }

response.end(JSON.stringify(mas));

response.end();

})

.catch(e => console.error(e.stack))

.then(() => console.log('Успішно'))

}

if (request.url === "/" && request.method === "GET") {

const filePath = "index.html";

fs.access(filePath, fs.constants.R\_OK, err => {

if (err) {

response.statusCode = 404;

response.end("Resourse not found!");

}

else {

fs.createReadStream(filePath).pipe(response);

}

});

}

}).listen(8081);

console.log('Server running at http://127.0.0.1:8081/');

Отже, я навчився використувати Promise.