**Luodaan backend wikisearch Node.js:llä**

1. Luo Visual Studio Codessa kansio **wiki**
2. Luo wiki-kansion sisälle kansio **backend** Terminal-ikkunassa komennolla **cd backend**
3. mene backend-kansion sisälle ja asenna **request-kirjasto** komennolla:

npm install –save request

1. Aja Terminal-ikkunassa npm init ja painele Enteriä, anna tiedostolle nimi **package.json**, niin **package.json** tiedosto luodaan. Voit antaa tietoja kysyttyihin asioihin, esim. **author**-kohtaan oman nimesi.

{

"name": "backend",

"version": "1.0.0",

"description": "",

"main": "app.js",

"dependencies": {

"express": "^4.17.1",

"request": "^2.88.2"

},

"devDependencies": {

"nodemon": "^2.0.3"

},

"scripts": {

"start": "nodemon api.js",

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "",

"license": "ISC"

}

1. Seuraavana luodaan backend-kansioon uusi tiedosto nimeltä **App.js** ja kirjoitetaan koodi **App.js**-tiedostoon

**App.js**

const request = require("request");

let haku = "";

if (process.argv[3] != null) {

haku = encodeURIComponent(process.argv[2] + " " + process.argv[3]);

haku.replace(" ", "%20");

} else {

haku = encodeURIComponent(process.argv[2]);

}

const wikisearch = (haku, callback) => {

const url =

"https://fi.wikipedia.org/w/api.php?action=query&titles=" +

haku +

"&format=json&formatversion=2&prop=extracts&exintro&explaintext&exsentences=4";

request({ uri: url, json: true }, (error, response, body) => {

if (error) {

callback("ei yhteyttä darkboxiin", undefined);

} else if (body.error) {

callback("Ei oo paikkaa", undefined);

} else {

callback(undefined, body.query.pages[0].extract);

}

});

};

wikisearch(haku, (error, data) => {

if (error) {

console.log(error);

} else {

console.log(data);

}

});

Rivi 1 tuodaan **request**-kirjasto käyttöön

Rivi 3 **let**-muuttajassa haku, joka on tyhjä

Rivi 4 Komentirivi parametriin päästää käsiksi **process-argv**. Komentoriviltä otetaan parametreinä sanat ja korvataan välilyönti **%20**:llä **(%20** on HTML-kielessä välilyönnin koodi)

**const wikisearch**-funktio: wikipedian hakuparametri, kutsuu sen API ja tehdään sille haku otsikoista, halutaan json-muodossa, versio 2. Palautetaan sisältö (**extract**)

Käytetään **request**-kirjastoa, annetaan url ja palautetaan API.

Jos tapahtui virheitä (**error**), annetaan virhe ilmoitus.

Jos ei saatu virhettä, mutta bodyssä (**body.error**) ei löytynyt mitään, annetaan virhe

jos kaikki meni ok, haetaan sivut ja niiden haettu data (**extract**)

Viimeisenä kutsuu wikisearch-funktiota

1. Ajetaan Terminal-ikkunassa komento: **node app.js koli**

Tämä Koli-teksti pitäisi näkyä Terminal-ikkunassa komennon jälkeen.

$ **node app.js koli**

Koli on Pohjois-Karjalan korkein vaara Pielisen rannalla Lieksassa. Koli on lähinnä valkoista kvartsiittia, ja sen laella on suuria puuttomia, paljaita valkoisia kalliopintoja. Kolin tärkeimmät huiput ovat Ukko- ja Akkakoli sekä Paha-Koli, jossa on jyrkkä seinämä. Kolin korkein kohta on Ukkokolilla.

Jos muokkaamme kokeeksi riviä

callback(undefined, body.query.pages[0].extract);

tähän muotoon

callback(undefined, body.query.pages);

nähdään **Terminal**-ikkunassa ajettaessa rakennetta:

[

{

**pageid**: 20761,

**ns**: 0,

**title**: 'Koli',

**extract**: 'Koli on Pohjois-Karjalan korkein vaara Pielisen rannalla Lieksassa. Koli on lähinnä valkoista kvartsiittia, ja sen laella on suuria puuttomia, paljaita valkoisia kalliopintoja. Kolin tärkeimmät huiput ovat Ukko- ja Akkakoli sekä Paha-Koli, jossa on jyrkkä seinämä. Kolin korkein kohta on Ukkokolilla.'

}

]

**TEHDÄÄN RAJAPINTA**

1. Seuraavaksi tehdään tästä **Node** API-rajapintaan. (Googlessa hakusana **node rest api**)

Nodelle löytyy paljon paketteja APIen rakentamiseen. Käytetyin on **Express**.

1. Asennetaan **Express**, anna Terminal-ikkunassa komento:

npm install express –save

1. Tehdään backend-kansioon **Api.js** niminen tiedosto

**backend/Api.js**

const express = require("express");

const app = express();

app.listen(4000, () => {

console.log("Server running on port 4000");

});

Rivi 1: hakee **Express**-kirjasto

Rivi 3: alustetaan Express applikaatio

Rivi 5: Kuuntelee porttia 4000

1. Testaa Terminal-ikkunassa komennolla **node api.js**, pitäisi vastata ”*Server running on port 4000”*

Määritellään endpointti, mikä vastaa meille, *Hello World!* Eli jos menisimme palvelimelle osoitteeseen /hello se vastasi meille Hello World!. Lisää alla oleva koodi **Api.js** -tiedostoon.

app.get("/hello", (req, res, next) => {

res.end("Hello world!");

});

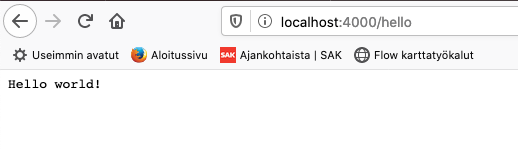
**req**, joka on pyyntö

**res** on vastaus

**next**-parametrillä voimme siirtyä eteenpäin mahdollisiin muihin endpointteihin (app.get)

**res.end** kertoo, kun lähetetään data, että tässä oli koko setti, hoida hommat.

1. Kokeillaan ensin Terminal-ikkunassa komennolla **node api.js**
2. Mene seuraavana internet selaimeen ja kirjoita osoiteriville **localhost:4000/hello**. Saadaan App toimintaan ja kuuntelemaan sekä toimimaan. Eli nyt meillä on API-rajapinta, jota me voidaan kutsua <url:n> päästä



1. Otetaan wikisearch-käyttöön. Mene **App.js**-tiedostoon ja lisää loppuu module.export…. (katso alla oleva tiedosto)

**backend/App.js**

const request = require("request");

let haku = "";

if (process.argv[3] != null) {

haku = encodeURIComponent(process.argv[2] + " " + process.argv[3]);

haku.replace(" ", "%20");

} else {

haku = encodeURIComponent(process.argv[2]);

}

const wikisearch = (haku, callback) => {

const url =

"https://fi.wikipedia.org/w/api.php?action=query&titles=" +

haku +

"&format=json&formatversion=2&prop=extracts&exintro&explaintext&exsentences=4";

request({ uri: url, json: true }, (error, response, body) => {

if (error) {

callback("ei yhteyttä darkboxiin", undefined);

} else if (body.error) {

callback("Ei oo paikkaa", undefined);

} else {

callback(undefined, body.query.pages[0].extract);

}

});

};

wikisearch(haku, (error, data) => {

if (error) {

console.log(error);

} else {

console.log(data);

}

});

module.exports ={

wikisearch,

};

Wikisearchillä on kaksi parametriä, toinen on **haku** ja toinen callback **(error, data)**

1. Mene **Api.js**-tiedostoon ja lisää rivi, jossa otetaan wikisearch- käyttöön App:sta

const {wikisearch} = require (”./wikisearch”);

**backend/Api.js**

const express = require("express");

const { wikisearch } = require("./wikisearch");

const app = express();

app.listen(4000, () => {

console.log("Server running on port 4000");

});

app.get("/hello", (req, res, next) => {

res.end("Hello world!");

});

1. Kopioidaan **App.js** tiedostosta alla oleva koodi ja liitetään se **Api.js**-tiedostoon

kopioi App.js:stä

wikisearch(haku, (error, data) => {

if (error) {

console.log(error);

} else {

console.log(data);

}

});

Api.js:ään ja muokkaa wikisearchissä **console.log**-kohdat 🡪 **res.end** ja **res.json**

**backend/Api.js**

const express = require("express");

const { wikisearch } = require("./wikisearch");

const app = express();

app.listen(4000, () => {

console.log("Server running on port 4000");

});

app.get("/hello", (req, res, next) => {

wikisearch(haku, (error, data) => {

if (error) {

res.end(error);

} else {

res.json(data);

}res

});

});

eli kun kutsutaan selaimessa **hello**, niin tehdään **wikisearch-haku.**

Jos tulee virhe, niin lähetetään se takaisin tai (else)

vaihtoehtoisesti jos saadaan dataa, ja se on json dataa, niin lähetetään json eteenpäin

1. Luo uusi tiedosto **wikisearch.js** ja siirretään **App.js**-tiedostosta wikisearch-funktio

**backend/wikisearch.js**

const wikisearch = (haku, callback) => {

const url =

"https://fi.wikipedia.org/w/api.php?action=query&titles=" +

haku +

"&format=json&formatversion=2&prop=extracts&exintro&explaintext&exsentences=4";

request({ uri: url, json: true }, (error, response, body) => {

if (error) {

callback("ei yhteyttä darkboxiin", undefined);

} else if (body.error) {

callback("Ei oo paikkaa", undefined);

} else {

callback(undefined, body.query.pages[0].extract);

}

});

};

Lisäksi tarvitaan lisätä **request**

const request = require("request");

eli nyt varsinainen haku on tässä **wikisearch.js**-tiedostossa, joka näyttää tältä

**backend/wikisearch.js**

const request = require("request");

const wikisearch = (haku, callback) => {

const url =

"https://fi.wikipedia.org/w/api.php?action=query&titles=" +

haku +

"&format=json&formatversion=2&prop=extracts&exintro&explaintext&exsentences=4";

request({ uri: url, json: true }, (error, response, body) => {

if (error) {

callback("ei yhteyttä darkboxiin", undefined);

} else if (body.error) {

callback("Ei oo paikkaa", undefined);

} else {

callback(undefined, body.query.pages[0].extract);

}

});

};

module.exports = {

wikisearch,

};

ja tässä vaiheessa **App.js** (muokkaa myös ylin **const** rivi)

**backend/App.js**

const { wikisearch } = require("./wikisearch");

let haku = "";

if (process.argv[3] != null) {

haku = encodeURIComponent(process.argv[2] + " " + process.argv[3]);

haku.replace(" ", "%20");

} else {

haku = encodeURIComponent(process.argv[2]);

}

wikisearch(haku, (error, data) => {

if (error) {

console.log(error);

} else {

console.log(data);

}

});

Nyt korjataan haku Api.js-tiedostoon

**backend/Api.js**

const express = require("express");

const { wikisearch } = require("./wikisearch");

const app = express();

app.listen(4000, () => {

console.log("Server running on port 4000");

});

app.get("/hello", (req, res, next) => {

wikisearch("joensuu", (error, data) => {

if (error) {

res.end(error);

} else {

res.json(data);

}

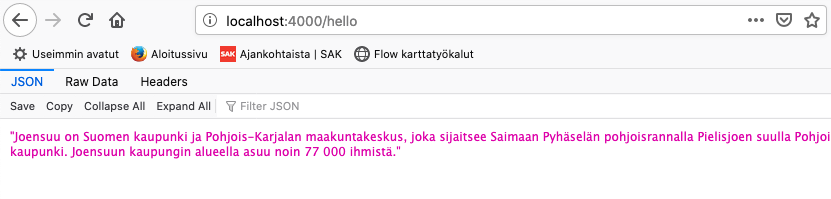
});

});

1. Testatataampa välillä

käynnistä Terminaalissa komennolla: **node api.js**, jolloin serveri käynnistyy

ja mene selaimeen, kirjoita osoiteriville **http://localhost:4000/hello**



1. Nyt jatketaan niin, että pystymme määrittelemään parametrin mitä halutaan haettavan (Nyt Joensuu annettu)

(**Lisätietoja**: Googleta 🡪 express parameter example)

Muokkaa **api.js (**app.get:iin lisää **/wiki** ja wikisearchiin lisää **reg.query.haku**)

const express = require("express");

const { wikisearch } = require("./wikisearch");

const app = express();

app.listen(4000, () => {

console.log("Server running on port 4000");

});

app.get("/wiki", (req, res, next) => {

wikisearch(req.query.haku, (error, data) => {

if (error) {

res.end(error);

} else {

res.json(data);

}

});

});

1. Asennetaan **nodemon**, jolla saadaan Nodessa käynnistetty automaattisesti muutoksen jälkeen Api.js:mme

npm install -g nodemon

**tai**

npm install nodemon

1. Aja ohjelma Terminal-ikkunassa komennolla:

nodemon api.js

1. Voi helpottaa starttaamista lisäämällä **package.json** tiedostoon scriptin **”start”: nodemon api.js**, ja jatkossa käynnistetään Terminal-ikkunassa komennolla: **npm start**

{

"name": "backend",

"version": "1.0.0",

"main": "app.js",

"scripts": {

"start": "nodemon api.js",

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "",

"license": "ISC",

"dependencies": {

"express": "^4.17.1",

"request": "^2.88.2"

},

"devDependencies": {

"nodemon": "^2.0.4"

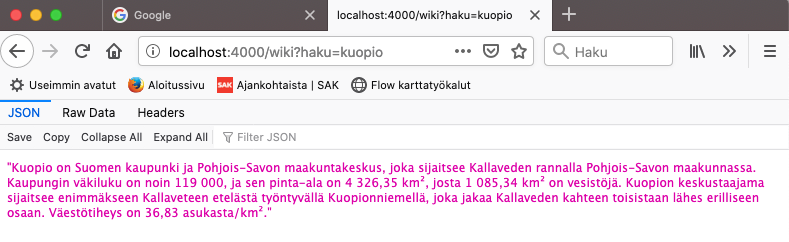
},

"description": ""

}

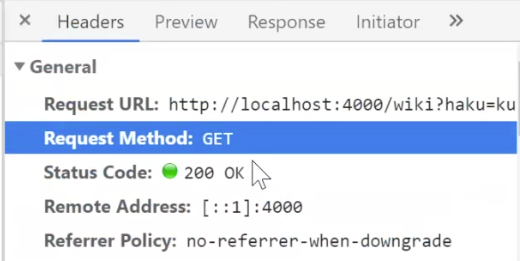
1. Testaa selaimessa, kirjoita osoiterivi

localhost:4000/wiki?haku=kuopio



1. Tutkitaampa tätä vielä selaimen puolella konsolissa, jonka saat näkyville painamalla F12

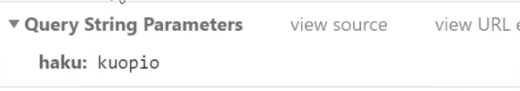
**GET**-tyyppinen kutsu, mitä lähetettiin (Request)



ja mitä saatiin takaisin (Response)



Mitkä parametrit oli



**Response** dataa voi katsella/tutkiskella **Preview**- ja **Response**-välilehdiltä

