A screenshot of a computer

Description automatically generatedA computer screen shot of a black screen

Description automatically generatedA computer screen with white text

Description automatically generated

**Lab2.js**

import yargs from "yargs";

import { hideBin } from "yargs/helpers";

import {

provinces,

fullNameAndProvincePromise,

transferPaymentsFromWebPromise,

transferPaymentForProvincePromise,

} from "./lab2\_routines.js";

const argv = yargs(hideBin(process.argv))

.options({

firstname: {

demandOption: true,

alias: "fname",

describe: "Resident's first name",

string: true,

},

lastname: {

demandOption: true,

alias: "lname",

describe: "Resident's last name",

string: true,

},

province: {

demandOption: true,

alias: "prov",

describe: "Resident's home province",

string: true,

choices: provinces.map(provinces=>provinces.code),

},

})

.help()

.alias("help", "h")

.parse();

let firstname = argv.firstname;

let lastname = argv.lastname;

let province = argv.province;

let fullNameProvince;

let gftAmount;

let gtfData;

//1st PROMISE

fullNameAndProvincePromise(firstname, lastname, province)

.then((results) => {

process.stdout.write(`${results.firstname}, ${results.lastname} lives in ${results.province}. `);

})

.catch((err) => {

console.log(`Error ==> ${err}`);

});

const prov = province.toLowerCase();

//2ND PROMISE

transferPaymentsFromWebPromise()

.then((response) => {

gtfData = response;

//3RD PROMISE

transferPaymentForProvincePromise(gtfData, prov)

.then((response) => {

console.log(`It received ${response} in transfer payments`);

})

.catch((err) => {

console.log("Cannot retrieve the price")

});

})

.catch((err) => {

console.log(`The json file does not exist`);

})

**Lab2\_routines.js**

import got from "got";

const provinces = [

{ code: "NS", name: "Nova Scotia" },

{ code: "NL", name: "Newfoundland" },

{ code: "NB", name: "New Brunswick" },

{ code: "PE", name: "Prince Edward Island" },

{ code: "QC", name: "Quebec" },

{ code: "ON", name: "Ontario" },

{ code: "MB", name: "Manitoba" },

{ code: "SK", name: "Saskatchewan" },

{ code: "AB", name: "Alberta" },

{ code: "BC", name: "British Columbia" },

{ code: "NT", name: "North West Territories" },

{ code: "NU", name: "Nunavut" },

{ code: "YT", name: "Yukon Territory" },

];

const FISCALYEAR = "2022-2023";

let provincesCodes = provinces.map(provinces => provinces.code);

// Create a currency formatter.

const currencyFormatter = (numberToFormat) =>

new Intl.NumberFormat("en-US", {

style: "currency",

currency: "USD",

minimumFractionDigits: 0,

}).format(numberToFormat);

// note weΓÇÖre using.then/.catch syntax here

const transferPaymentsFromWebPromise = () => {

let srcAddr =

"http://www.infrastructure.gc.ca/alt-format/opendata/transfer-program-programmes-de-transfert-bil.json";

return new Promise((resolve, reject) => {

got(srcAddr, { responseType: "json" })

.then((response) => {

resolve(response.body.gtf);

})

.catch((err) => {

console.log(`Error ==> ${err}`);

reject(err);

});

});

};

const transferPaymentForProvincePromise = (gtfData, provCode) => {

return new Promise((resolve, reject) => {

let value = gtfData[provCode][FISCALYEAR];

if(value != null) {

resolve(currencyFormatter(value));

}else{

reject(err);

}

});

}

// note weΓÇÖre using.then/.catch syntax here

const fullNameAndProvincePromise = (firstname, lastname, province) => {

return new Promise((resolve, reject) => {

if(provinces.map(provinces => provinces.code).includes(province)){

resolve({firstname, lastname, province});

}else{

reject("Province not found");

}

});

};

export {

provinces,

currencyFormatter,

fullNameAndProvincePromise,

transferPaymentsFromWebPromise,

transferPaymentForProvincePromise,

};