SOFTWARE EXAM v.6 [NODE JS]

Applicant: Jose A. Perez Jr.

WEBCAST TECHNOLOGIES, INC

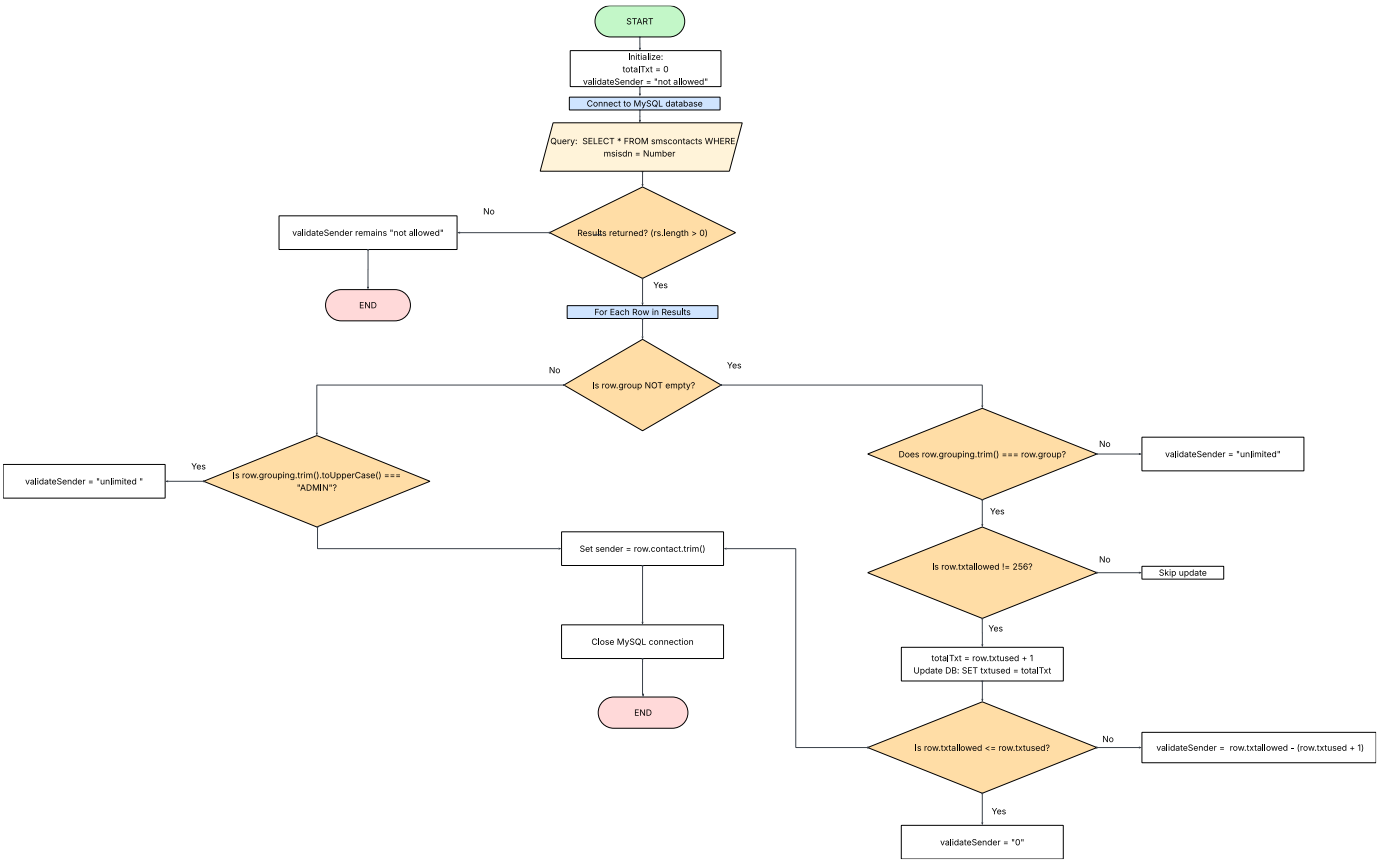
Applying to the position of BACKEND DEVELOPER

1. Make a Flowchart for the code/function below

```
let totalTxt = 0;
const mysql = require("mysql");
var connection = mysql.createConnection({
        host    : 'localhost',
        user    : 'dbuser',
        password : 'secret',
        database : 'my_db'
});

connection.connect();
let validateSender = "not allowed";

const rs = connection.query(` SELECT * FROM smscontacts WHERE msisdn = ${Number}` ,
 function (error, results, fields) {
  if (error) throw error;
  return results;
 }
);

if(rs.length){
```

```
  rs.forEach(row => {
    if(row.group !== ""){
      if(row.grouping.trim() === row.group){
        if(row.txtallowed !== 256){
          totalTxt = row.txtused + 1;
          connection.query(` UPDATE smscontacts SET txtused = ${totalTxt} WHERE _msisdn =
${Number}` );
        }
        if(row.txtallowed <= row.txtused){
          validateSender = "0";
        } else {
          validateSender = row.txtallowed - (row.txtused + 1);
        }
      } else {
        validateSender = "unlimited";
      }
    } else if (row.grouping.trim().toUpperCase() === 'ADMIN'){
      validateSender = "unlimited";
    } else if (row.group.trim().toUpperCase() === 'ADMIN'){
      validateSender = "unlimited";
    }
    sender = row.contact.trim();
  });
}
connection.end();
```

ANSWER:



[diagram link]

2. ES5 to ES6. Write ff. ES5 to ES6 standard

QUESTION a:

a. function greet(name) {
   return 'hello' + name;
   }

ANSWER:

```
const greet = (name) => {
 return `hello ${name}`;
};
```

QUESTION b:

b. var obj1 = {a:1, b:2, c:3, d:4}

var a = obj1.a

var b = obj1.b

var c = obj1.c

var d = obj1.d

ANSWER:

```
const obj1 = {a: 1, b: 2, c: 3, d: 4};

const {a, b, c, d} = obj1;
```

QUESTION c:

c. function isLesser (a, b, callback) {
    var lesser = false
    if (a < b) {
        lesser = true
    }
    callback(lesser)
}

ANSWER:

```
const isLesser = (a, b, callback) => {

    let lesser = false;

    if (a < b) {

        lesser = true;

    }

    callback(lesser);

};
```

QUESTION d:

d. function isLesser(a, b, callback) {
    var lesser = false;
    if (a < b) {
        lesser = true
    }

```
        callback(lesser)
        }
```

ANSWER:

```
const isLesser = (a, b, callback) => {
    let lesser = false;
    if (a < b) {
        lesser = true;
    }
    callback(lesser);
}
```

3. Write a nodeJS function that has the following rules.
    a. Function name is addToList
    b. It accepts a parameter in Array type. Parameter name is newList
    c. It has a global variable named globalList of type Array
    d. The function will add the items from the newList to globalList if the item is not yet existing in the globalList

ANSWER:

```
// declare variable
let globalList = [];


function addToList(newList) {
  if (!Array.isArray(newList)) {
    throw new Error("Parameter must be an array");
  }

  newList.forEach(item => {
    if (!globalList.includes(item)) {
      globalList.push(item);
```

```
    }
  });
}
// call function
addToList([1, 2, 3]);
console.log(globalList);
```

OUTPUT:

```
$ node exam3.js
[1, 2, 3]
```

4. Write a nodeJS script that has the following rules
    a. Has a global variable
        i. ADD – value is either true or false
        ii. MULTIPLY – value is either true or false
    b. Has a function named compute
        i. This function is an asynchronous
        ii. This function accepts an array of numbers
        iii. If the global variable ADD is true, this function will call another function named addNumbers
        iv. If the global variable MULTIPLY is true, this function will call another function named multiplyNumbers
        v. The multiplyNumbers function will only be called after the addNumbers process is done
    c. Has a function named addNumbers
        i. This function accepts an array of numbers
        ii. It will log on the screen the sum of the numbers on the array list
    d. Has a function named multiplyNumbers
        i. This function accepts an array of numbers
        ii. It will log on the screen the product of the numbers on the array list

ANSWER:

```
const ADD = true;
const MULTIPLY = true;
```

```
// compute function
async function compute(numbers) {
  if (ADD) {
    await addNumbers(numbers);
  }

  if (MULTIPLY) {
    await multiplyNumbers(numbers);
  }
}

// addNumbers function
function addNumbers(numbers) {
  return new Promise((resolve) => {
    const sum = numbers.reduce((acc, num) => acc + num, 0);
    console.log("Sum:", sum);
    resolve();
  });
}

// multiplyNumbers function
function multiplyNumbers(numbers) {
  return new Promise((resolve) => {
    const product = numbers.reduce((acc, num) => acc * num, 1);
    console.log("Product:", product);
```

```
  resolve();
 });
}

// call async function
compute([1, 2, 3, 4]);
```

OUTPUT:

```
$ node exam4.js
Sum: 10
Product: 24
```