Introduction

This is a test that is designed to check if you are able to produce a working Node.JS server. You will find a mix of questions and one exercise.

Questions

Please reply to the following questions:

1) What is the result of this code in the console

```
for (var i = 0; i < 100; i++) {
    setTimeout(function() {
        console.log(i);
    }, 200);
}
```

2) What is the result of this code in the console

```
(async function() {
    function waitForMe() {
        return new Promise(function(resolve, reject) {
            setTimeout(function() {
                resolve("hello");
            }, 200);
        });
    }

    const result = await waitForMe();
    console.log(result);
})();
```

3) Explain the difference

Explain the difference between those:

```
var a;
let b;
const c;
```

4) What is the result of this code in the console

We got a module "test.js":

```
var arr = [];

module.exports = arr;
```

We got a module using it:

```
var test = require("./test");

test.push("hello");

console.log(JSON.stringify(test));
```

We got another module using it later:

```
var test = require("./test");

test.push("another");

console.log(JSON.stringify(test));
```

Exercise

For this part, your goal is to produce a working Node.JS server.

Goals

- Produce a working code.
- Structure the project the best way you can (including README.md for ex.).
- Produce an easy installer using npm, the README.md should include every needed step to install and start running the code.
- Add a TODO list with further improvements and an outline of what you would do.

Don't hesitate to add comment to your code, test it, anything that you feel is necessary. Our target is also to see if you can produce everything surround the project that is **needed for the team to understand it** and other parties to deploy and monitor it.

You are free to use any version of Node.JS, and any version of Postgres (we recommend you indicate inside the README.md file which version of each program you support).

Statement

Assume we are a software company that wants to accept payments on our system, using credit cards.

So far, we limit ourselves to the US market as a first step (so we don't care about translation, or various types of credit card out there, and everything has to be in english -code and user interface-), our product is ready, all we miss is the payement gateway.

A credit card has the following information that we need to retrieve:

- A 16 digits number,
- A 3 digits CVV,
- A card holder name,
- An expiration date (month and year).

Your goal is to implement, using Node.JS / express and postgres (and any other framework you may want to use) the following elements:

- A page that let user enter credit card's information.
- An ajax call from this page, that take the user input, send it, validate it, store everything into database, and return a message indicating if everything went well or not to the client (The card number, the CVV must be stored encrypted into the database).
- Then, the page show a popup to the client if it has worked or not.

For the sake of this exercise we don't really care about security in general:

- We assume the user is already logged in and everything is fine about this in general.
- About the encryption on CVV and 16 digits, you can use third party library, and you are free to choose which algorithm to use (explain why you choose it in the README.md file).

Once this is done, extend the code to add the Luhn Algorithm to the code base. You are free to choose how to add it, and explain why you choose this or that approach in the README.md.

Submit the code

Once you're ready, you can choose between one of them:

- -OR- Create a repository on github/bitbucket/any git hoster,
- -OR- Zip the project (remove node_modules from the zip please),
- -OR- Any FTP or storage where I can download from there.

No matter what you choose, send me an email with the link to download the project to charles.villette@carma.com, we will come back shortly to you later.