

HUMAN CARE SYSTEMS

INTERVIEW PROCESS

Back-End Challenge

Author:

Alejandro RODRÍGUEZ RECHE

MONTEVIDEO, URUGUAY
Wednesday, December 8th, 2021.

Contents

1	Initial installations	2
2	Packages needed	2
3	Configuration needed	2
4	Solution steps	4
a	Parse CSV file	4
b	Work on Patients logic to insert them to the database	4
c	Work on Emails logic to create emails as requested	5
d	Work on logic for adding created emails to the database	5
e	Create a test file and write requested tests there	6

1 Initial installations

Several installations were needed at first, for example:

- NVM for Windows.
- Node.js and NPM.
- MongoDB

2 Packages needed

In order to properly use different tools, some extra NPM packages were needed, like:

- **csv-parser**, a package for parsing CSV files and retrieving data from them.
- **mongod** for handling interactions with MongoDB databases from a Node.js project.
- **jest** for unit testing of the code, in order to make the proper tests as requested.

3 Configuration needed

After installing **jest**, it was needed to set **"test": "jest"** in **package.json** file.

```
{ } package.json ×
{ } package.json > { } scripts > test
1  {
2    "name": "backend-challenge",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "jest"
8    },
9    "author": "",
10   "license": "ISC",
11   "dependencies": {
12     "csv-parser": "^3.0.0",
13     "mongodb": "^4.2.1"
14   },
15   "devDependencies": {
16     "jest": "^27.4.3"
17   }
18 }
19
```

Figure 1: package.json file.

4 Solution steps

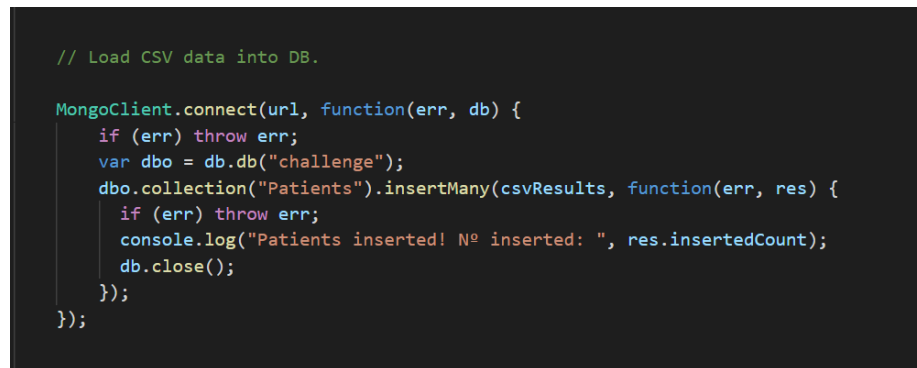
a Parse CSV file



```
JS main.js x
JS main.js > ...
1  const csv = require('csv-parser');
2  const fs = require('fs');
3  const MongoClient = require('mongodb').MongoClient;
4  var url = "mongodb://localhost:27017/challenge";
5
6  var csvResults = [];
7  var emailsToLoad = [];
8
9  fs.createReadStream('data.csv')
10     .pipe(csv({ separator: '|' }))
11     .on('data', (data) => csvResults.push(data))
12     .on('end', () => {
13         // ... (rest of the code) ...
14     });
```

Figure 2: CSV file parsing with proper package import.

b Work on Patients logic to insert them to the database



```
// Load CSV data into DB.
MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("challenge");
  dbo.collection("Patients").insertMany(csvResults, function(err, res) {
    if (err) throw err;
    console.log("Patients inserted! Nº inserted: ", res.insertedCount);
    db.close();
  });
});
```

c Work on Emails logic to create emails as requested

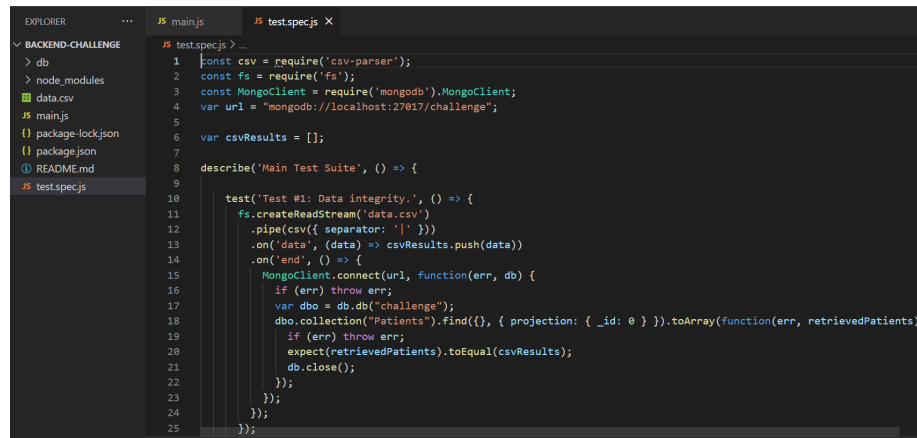
```
for (let index = 0; index < csvResults.length; index++) {
  const patient = csvResults[index];
  if (patient['CONSENT'] === 'Y') {
    let currentDate = new Date();
    let emails = [
      {
        'Name': "Day 1",
        'scheduled_date': currentDate.getFullYear() + '-' + currentDate.getMonth() + '-' + (currentDate.getDate() + 1),
        'email': patient['Email Address']
      },
      {
        'Name': "Day 2",
        'scheduled_date': currentDate.getFullYear() + '-' + currentDate.getMonth() + '-' + (currentDate.getDate() + 2),
        'email': patient['Email Address']
      },
      {
        'Name': "Day 3",
        'scheduled_date': currentDate.getFullYear() + '-' + currentDate.getMonth() + '-' + (currentDate.getDate() + 3),
        'email': patient['Email Address']
      },
      {
        'Name': "Day 4",
        'scheduled_date': currentDate.getFullYear() + '-' + currentDate.getMonth() + '-' + (currentDate.getDate() + 4),
        'email': patient['Email Address']
      }
    ];
  }
}
```

d Work on logic for adding created emails to the database

```
// Load into Emails db collection.

MongoClient.connect(url, function(err, db) {
  if (err) throw err;
  var dbo = db.db("challenge");
  dbo.collection("Emails").insertMany(emailsToLoad, function(err, res) {
    if (err) throw err;
    console.log("Emails inserted! Nº inserted: ", res.insertedCount);
    db.close();
  });
});
```

e Create a test file and write requested tests there



```
1 const csv = require('csv-parser');
2 const fs = require('fs');
3 const MongoClient = require('mongodb').MongoClient;
4 var url = "mongodb://localhost:27017/challenge";
5
6 var csvResults = [];
7
8 describe('Main Test Suite', () => {
9
10   test('Test #1: Data integrity.', () => {
11     fs.createReadStream('data.csv')
12       .pipe(csv({ separator: '|' })))
13       .on('data', (data) => csvResults.push(data))
14       .on('end', () => {
15         MongoClient.connect(url, function(err, db) {
16           if (err) throw err;
17           var dbo = db.db("challenge");
18           dbo.collection("Patients").find({}, { projection: { _id: 0 } }).toArray(function(err, retrievedPatients) {
19             if (err) throw err;
20             expect(retrievedPatients).toEqual(csvResults);
21             db.close();
22           });
23         });
24       });
25     });
26   });
27 });
```

Figure 3: Tests run with **npm run test** command in terminal.