KARIYA RAJ 21162101011

MICROSERVICES

PRACTICAL -6

AIM: To Implement the file system and its operation with NodeJS: "A-1" grocery shop owner wants to manage shop items using asynchronous coding technique of node and wants to perform following task:

As for the first four tasks for the grocery shop owner. I have decided 3 rows for the CRUD operations.

1. id
2. prod_name
3. price

TASK 6.1 : Reading Data From csv.

task_6_1.js :

```
const csv = require('csv-parser');
const fs=require('fs')
const filepath = './grocery.csv'
var readStream = fs.createReadStream(filepath);
readStream.pipe(csv())
.on('data', (row)=> {
console.log(row);
});
readStream.on('end', function(end) {
console.log("file read successfully")
});
```

```
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
node readFile.js
{ Id: '1', 'Product Name': 'Balaji Wafers', Price: '20' }
{ Id: '2', 'Product Name': 'Oreo Biscuits', Price: '30' }
file read successfully
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
```

TASK 6.2 : Adding Data to CSV.

Add_data.js:

```
const createCsvWriter = require('csv-writer').createObjectCsvWriter;
const csvWriter = createCsvWriter({
path: 'grocery.csv',
header: [
{ id: 'id', title: 'Id' },
{ id: 'prod_name', title: 'Product Name' },
{ id: 'price', title: 'Price' },
]
}
);
const data =
[
{
id: '1',
prod_name: 'Balaji Wafers',
price: 20,
},
{
id: '2',
prod_name: 'Oreo Biscuits',
price: 30,
},
]
csvWriter
.writeRecords(data)
.then(() => console.log('csv writen'));
```

OUTPUT:

```
found 0 vulnerabilities
 ~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
node add_data
csv writen
 ~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
```

TASK 6.3 : Deleting data from csv.

deleteData.js:

```
const fs = require('fs');
const csv = require('csv-parser');
const createCsvWriter = require('csv-writer').createObjectCsvWriter;

const filePath = 'grocery.csv';
const rowToDelete = 1;

const rows = [];

fs.createReadStream(filePath)
.pipe(csv())
.on('data', (row) => {
rows.push(row);
})
.on('end', () => {
if (rowToDelete >= 0 && rowToDelete < rows.length) {
rows.splice(rowToDelete, 1);
const csvWriter = createCsvWriter({
path: 'grocery.csv',
header: Object.keys(rows[0]).map(column => ({ id: column, title: column }))
});
csvWriter.writeRecords(rows)
.then(() => {
console.log('Row deleted successfully');
})
.catch((error) => {
console.error('Error deleting row:', error);
});
} else {
console.error('Invalid row index');
}
```

```
});
```

Output:

```
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
node deleteData
Row deleted successfully
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
node readFile
{ Id: '1', 'Product Name': 'Balaji Wafers', Price: '20' }
file read successfully
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
```

TASK 6.4 : Renaming the csv file.

deleteData.js:

```
const fs = require('fs');

const oldFilePath = 'grocery.csv';
const newFilePath = 'grocery1.csv';

fs.rename(oldFilePath, newFilePath, (error) => {
if (error) {
console.error('Error renaming file:', error);
} else {
console.log('File renamed successfully');
console.log('The new file name is ' + newFilePath);
}
});
```

Output:

```
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
node renameCsv.js
File renamed successfully
The new file name is grocery1.csv
~/Desktop/B.TECH CSE/SEM-5/MICROSERVICES/PRACTICALS/6
```

Task 6.5 : Create an application to manage the students grade sheet using csv file. Columns include Student name, Quiz_Marks, Mid-term_Marks,Assignment_Marks, final_exam_marksTotal_marks.

```
const express = require("express");
const csv = require("csv-parser");
const createCsvWriter =
require('csv-writer').createObjectCsvWriter
;

const fs = require("fs");
const filepath = "./students.csv";
const app = express();
app.use(express.json());
var readStream =
fs.createReadStream(filepath);
var users = [];

app.get("/students", (req, res) => {
readStream.pipe(csv()).on("data", (row) =>
{
users.push(row);
console.log(row);
```

```
});
readStream.on("end", function (end) {
console.log("file read successfully");
res.json(users);
});
// console.log(users);
// res.json(users);
});
//
-----------------------------------------------
-----------------------------------------------
--

const csvWriter = createCsvWriter({
path: "students.csv",
header: [
{ id: "sname", title: "Student_Name" },
{ id: "mmarks", title: "MidTerm_Marks" },
{ id: "amarks", title: "Assignment_Marks"
},
{ id: "fmarks", title: "Final_Marks" },
```

```
{ id: "tmarks", title: "Total_Marks" },
],
append: true,
});

app.post("/students", (req, res) => {
const data = req.body;
const createCsvWriter =
require("csv-writer").createObjectCsvWriter
;
const csvWriter = createCsvWriter({
path: "students.csv",
header: [
{ id: "sname", title: "Student_Name" },
{ id: "mmarks", title: "MidTerm_Marks" },
{ id: "amarks", title: "Assignment_Marks"
},
{ id: "fmarks", title: "Final_Marks" },
{ id: "tmarks", title: "Total_Marks" },
],
append:true,
```

```
});
if (
!data.sname ||
!data.mmarks ||
!data.amarks ||
!data.fmarks ||
!data.tmarks
) {
res.json({
message:
"All fields are mandatory please provide
all the required fields",
});
} else {
const x = [
{
sname: data.sname,
mmarks: data.mmarks,
amarks: data.amarks,
fmarks: data.fmarks,
tmarks: data.tmarks,
```

```
},
];
csvWriter.writeRecords(x).then(() =>
console.log("csv writen"));
}
res.json({ message: "Data has been
successfully added in the csv file." });
});

//
---------------------------------------------
---------------------------------------------
--------------

app.post("/delete", (req, res) => {
const no_rows=req.body.rows;
if(!no_rows){
res.json({message:"No of rows required"});
}else{
const rows = [];
fs.createReadStream(filepath)
```

```
.pipe(csv())
.on('data', (row) => {
rows.push(row);
})
.on('end', () => {
if (no_rows >= 0 && no_rows < rows.length)
{
rows.splice(no_rows, 1);
const csvWriter = createCsvWriter({
path: 'grocery.csv',
header: Object.keys(rows[0]).map(column =>
({ id: column, title: column }))
});
csvWriter.writeRecords(rows)
.then(() => {
console.log('Row deleted successfully');
})
.catch((error) => {
console.error('Error deleting row:',
error);
});
```

```
} else {
console.error('Invalid row index');
}
});

res.json({message:"Successfullt deleted
rows."});
}
});

app.post("/rename", (req, res) => {
const d = req.body.name;
if (!d) {
res.json({ message: "Name is required for
renaming the file." });
} else {
const oldFilePath = "students.csv";
fs.rename(oldFilePath, d, (error) => {
if (error) {
console.error("Error renaming file:",
error);
```

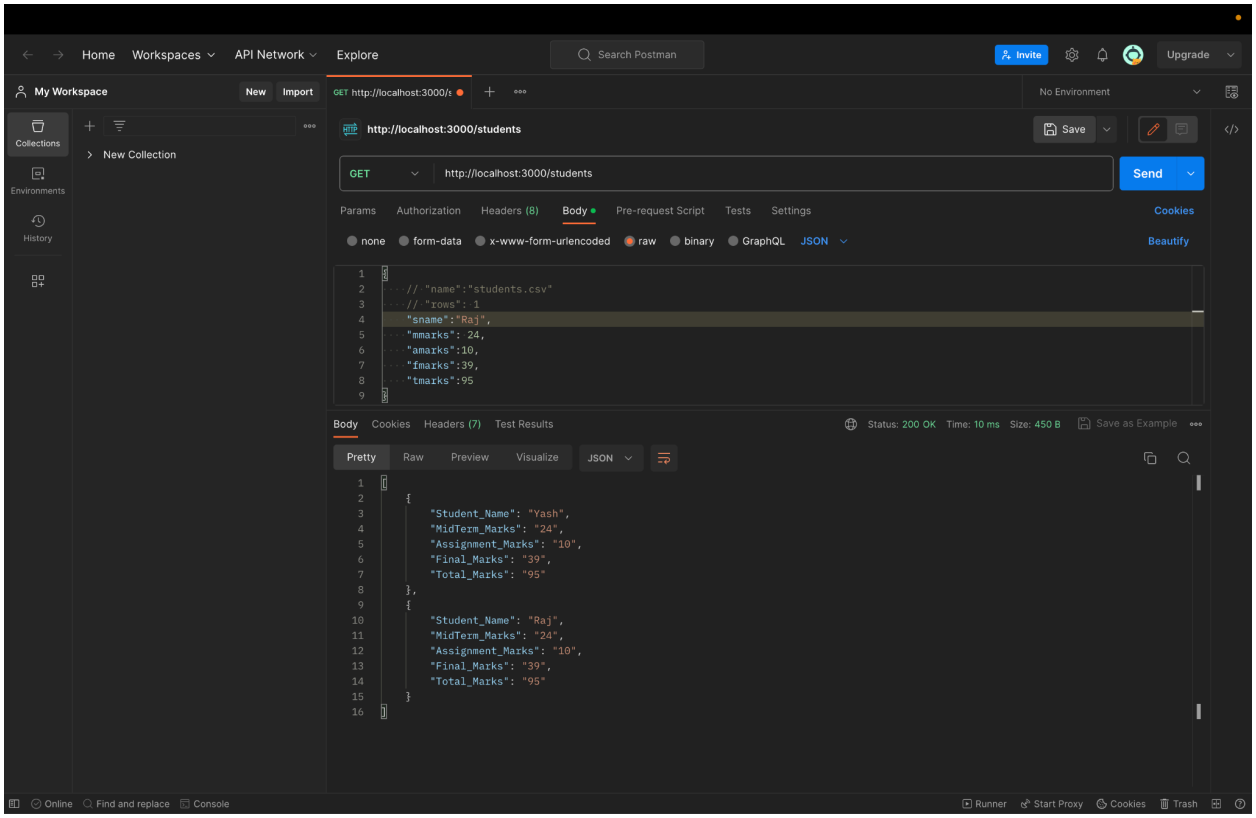
```
} else {
console.log("File renamed successfully");
console.log("The new file name is " + d);
}
});
res.json({message:"Successfully changed the
name of the file"});
}
});

app.listen(3000, () => {
console.log("Server is running on port
3000");
});
```

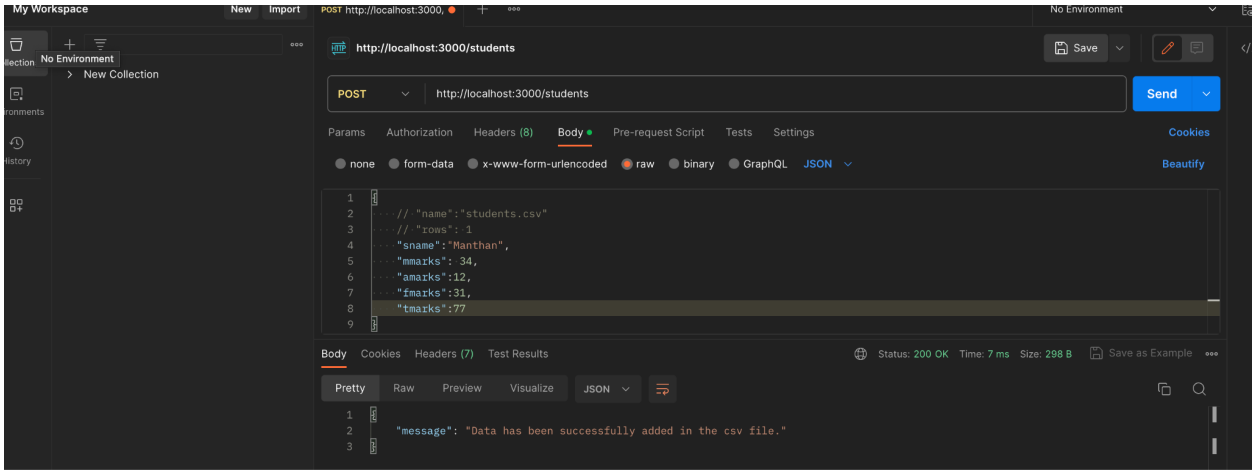
OUTPUT:

GET REQUEST:

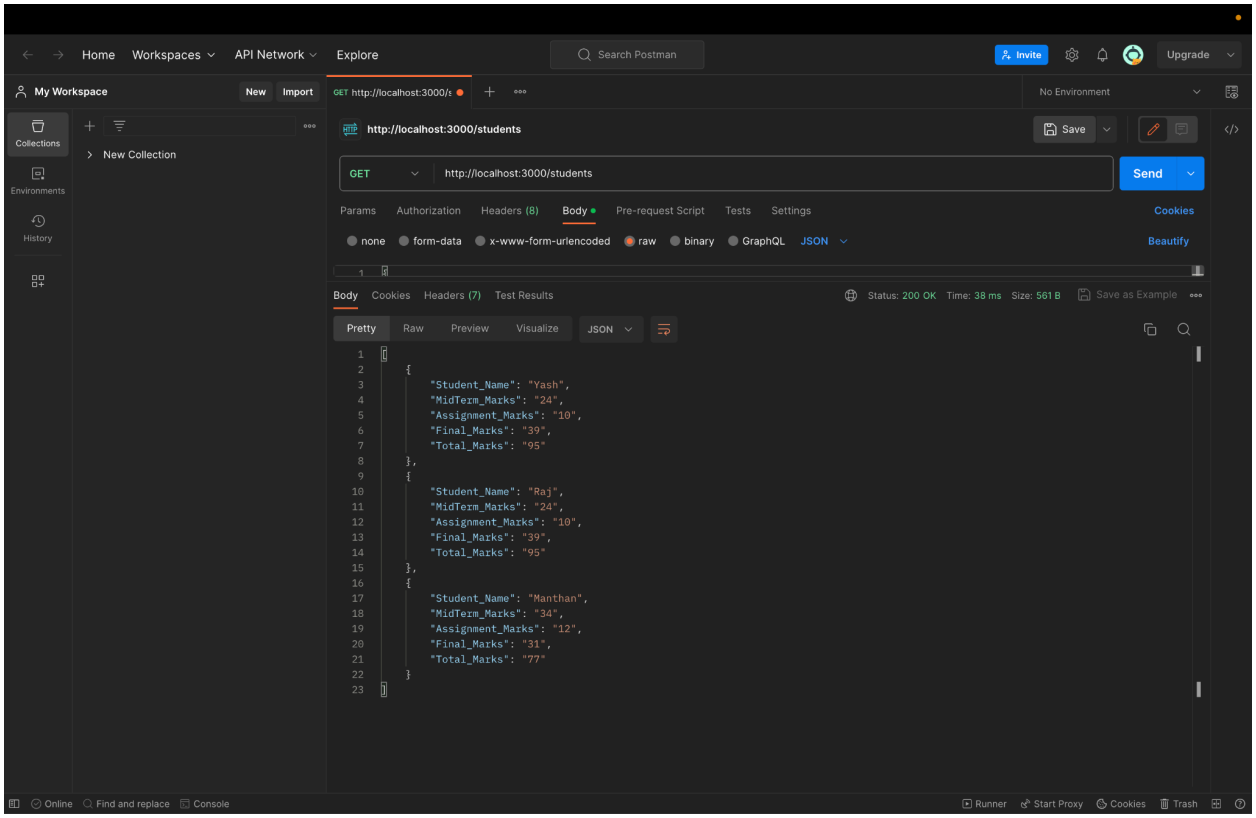
KARIYA RAJ 21162101011



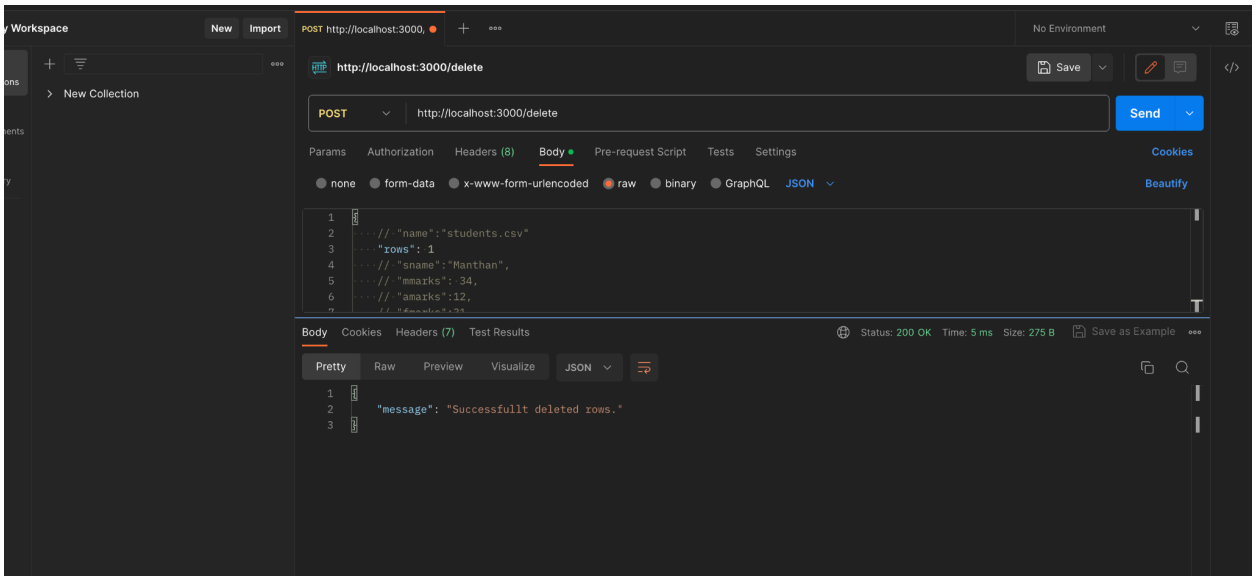
POST REQUEST:



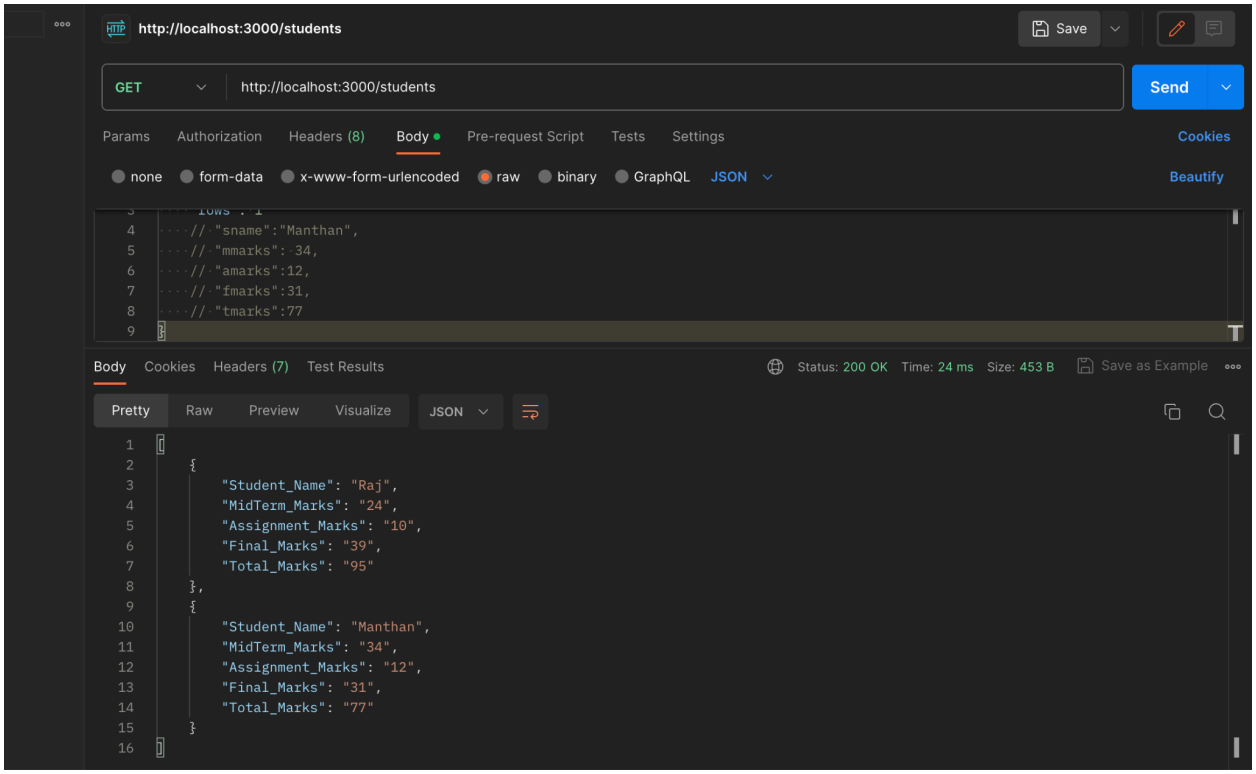
KARIYA RAJ 21162101011



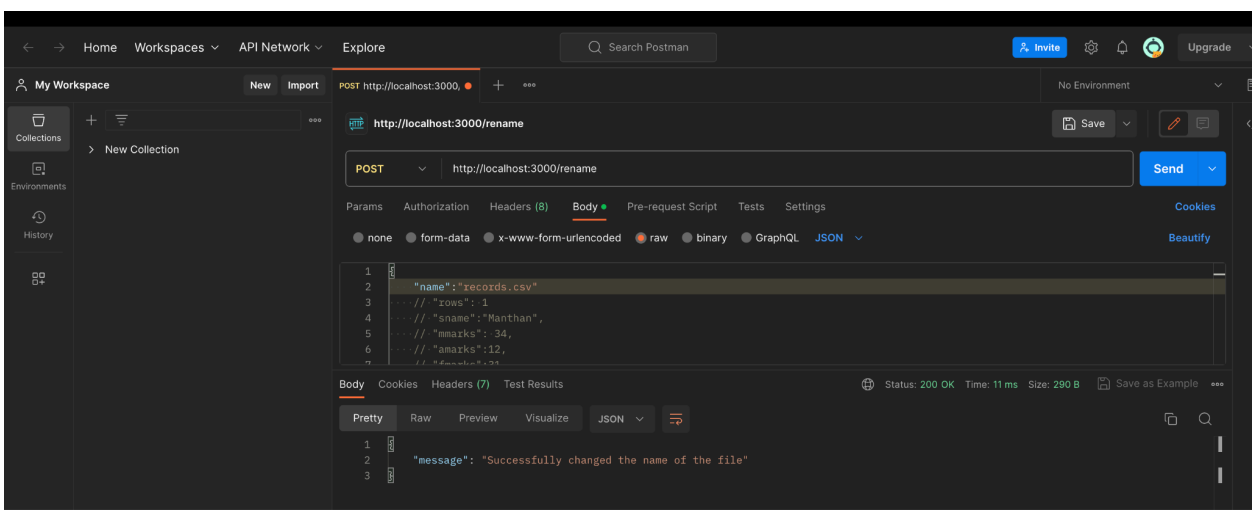
DELETE REQUEST:



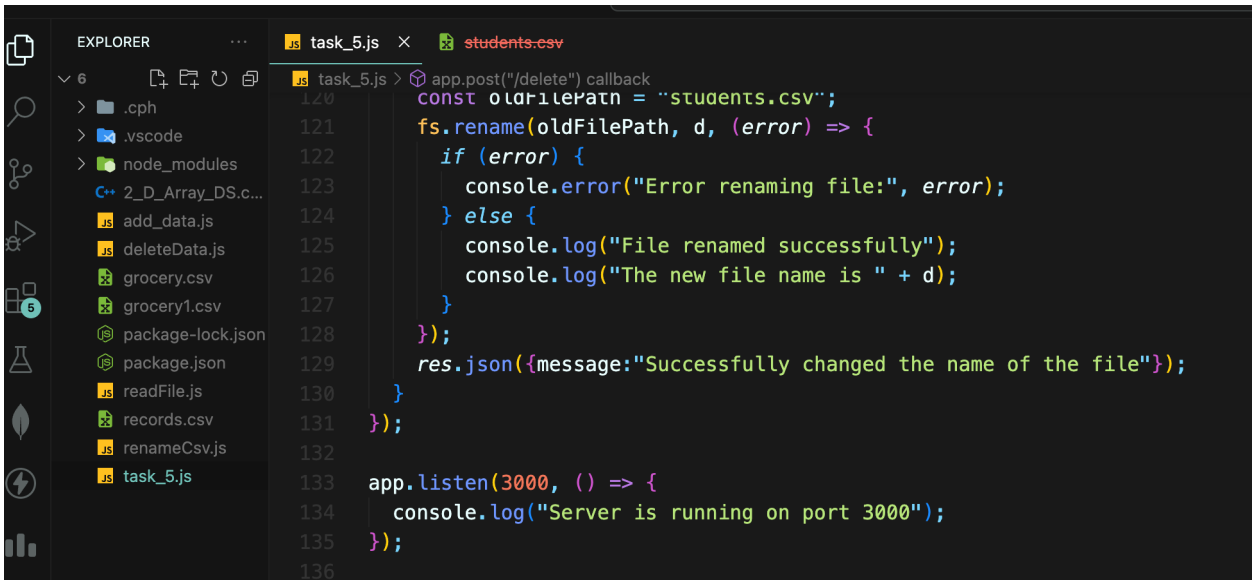
KARIYA RAJ 21162101011



Rename request:



KARIYA RAJ 21162101011




```
    const oldFilePath = "students.csv";
    fs.rename(oldFilePath, d, (error) => {
      if (error) {
        console.error("Error renaming file:", error);
      } else {
        console.log("File renamed successfully");
        console.log("The new file name is " + d);
      }
    });
    res.json({message:"Successfully changed the name of the file"});
  }
});

app.listen(3000, () => {
  console.log("Server is running on port 3000");
});
```