MODULE 6 : TASK 1

```
const express = require('express');
const app = express();
app.use(express.json());


let students = [
    {
        "name" : "alan",
        "age" : 22,
        "dept" : "cse",
        "active" : false
    },
    {
        "name" : "amal",
        "age" : 22,
        "dept" : "cse",
        "active" : false
    },
    {
        "name": "john",
        "age": 23,
        "dept": "cse",
        "active": true
    }
];

app.get('/',(req,res)=>{
    res.send('hello world this is a change.');
});

app.get('/sumoftwo',(req,res)=>{

   let num1 = parseInt(req.query.num1);
   let num2 = parseInt(req.query.num2)
   let sum = num1 + num2
    res.send(`SUM OF NUMBERS ${num1} + ${num2} is  ${sum}`);
});

app.get('/getStudents',(req,res)=>{
    res.json(students);
});
```

```
app.post('/insertstudent',(req,res)=>{

    let { name, age, dept, active } = req.body;

    if(name && age && dept && active){
        let convertedStudentName = name.toLowerCase().trim();
        let convertedStudentDept = dept.toLowerCase();

        let existing = students.find(each=> each.name === 
convertedStudentName);

        if(existing){
            res.send(`Student ${name} already exists.`);
        }
        else{
            let newStudent = { name: convertedStudentName, age , dept: 
convertedStudentDept, active};
            students.push(newStudent);
            res.send(`inserted student ${name} success.`);
        }

    }else{
        res.send('Please provide all necessary fields(name, age, dept, 
active).');
    }

})

app.delete('/deletestudent',(req,res)=>{
    let studentName = req.query.name;
    if(studentName){
        let convertedStudentName = studentName.toLowerCase().trim();

        const indexOfconvertedStudent = students.findIndex(each=> 
each.name.toLowerCase() === convertedStudentName)

        if(indexOfconvertedStudent >= 0){
            students.splice(indexOfconvertedStudent,1)
            res.send(`Student ${studentName} has been deleted 
successfully`);         }
        else{
            res.send(`Student ${studentName} not found.`);
        }

    }else{
        res.send('Please enter a valid name to delete');
    }
});
```

```
app.put('/updatestudent', (req,res)=>{
    let name = req.query.name;
    let newData = req.body;

    if(name){
        const index = students.findIndex(each=> each.name.toLowerCase() ===
name.toLowerCase());

    if(index >= 0){
        students[index] = {
            name: newData.name ||students[index].name,
            age: newData.age || students[index].age,
            dept: newData.dept || students[index].dept,
            active: newData.active !== undefined ? newData.active :
students[index].active
        };
        res.json({
            message: `Updated ${name} successfully`,
            updatedStudent: students[index]
        });
    }
    else{
        res.json({ message: `Student ${name} not found` });
    }

    }else{
        res.send('provide a name to update');
    }
});



app.listen(3200);
```

ACHIEVED CRUD OPREATIONS USING API CALLS.

Task 3

- Combine the fs library and store the student details for all the crud operations.

```
const express = require('express');
const app = express();
app.use(express.json());
const fs = require('fs');

const dbPath = 'db.json'

app.get('/getStudents', (req, res) => {
    const students = readStudentsFromFile();
    res.json(students);
    console.log('Read file db.json');
});

function readStudentsFromFile() {
    try {
        const data = fs.readFileSync(dbPath, 'utf-8');
        console.log('file readed.');
        return JSON.parse(data);
    } catch (err) {
        console.error('Error reading db.json:', err);
        return [];
    }
}


function writeStudentsToFile(students) {
    try {
        fs.writeFileSync(dbPath, JSON.stringify(students, null, 2));
        console.log('file writed.');
    } catch (err) {
        console.error('Error writing to db.json:', err);
    }
}


app.post('/insertstudent', (req, res) => {
    let { name, age, dept, active } = req.body;

    if (name && age && dept && active) {
        let convertedStudentName = name.toLowerCase().trim();
        let convertedStudentDept = dept.toLowerCase();

        const students = readStudentsFromFile();

        let existing = students.find(each => each.name ===
convertedStudentName);

        if (existing) {
            res.send(`Student ${name} already exists.`);
```

```
        } else {
            let newStudent = { name: convertedStudentName, age, dept:
convertedStudentDept, active };
            students.push(newStudent);
            writeStudentsToFile(students);
            res.send(`Inserted student ${name} successfully.`);
        }

    } else {
        res.send('Please provide all necessary fields(name, age, dept,
active).');
    }
});


app.delete('/deletestudent', (req, res) => {
    let studentName = req.query.name;

    if (studentName) {
        let convertedStudentName = studentName.toLowerCase().trim();

        const students = readStudentsFromFile();
        const index = students.findIndex(each => each.name.toLowerCase() ===
convertedStudentName);

        if (index >= 0) {
            students.splice(index, 1);
            writeStudentsToFile(students);
            res.send(`Student ${studentName} has been deleted successfully.`);
        } else {
            res.send(`Student ${studentName} not found.`);
        }

    } else {
        res.send('Please enter a valid name to delete.');
    }
});


app.put('/updatestudent', (req, res) => {
    let name = req.query.name;
    let newData = req.body;

    if (name) {
        const students = readStudentsFromFile();
        const index = students.findIndex(each => each.name.toLowerCase() ===
name.toLowerCase());

        if (index >= 0) {
```

```
            students[index] = {
                name: newData.name || students[index].name,
                age: newData.age || students[index].age,
                dept: newData.dept || students[index].dept,
                active: newData.active !== undefined ? newData.active :
students[index].active
            };
            writeStudentsToFile(students);
            res.json({
                message: `Updated ${name} successfully.`,
                updatedStudent: students[index]
            });
        } else {
            res.json({ message: `Student ${name} not found.` });
        }
    } else {
        res.send('Please provide a name to update.');
    }
});


app.listen(3200);
```