**Program 10:** Develop an express web application that can interact with REST API to perform CRUD operations on student data. (Use Postman)

**Aim:** Need to develop a web application using ExpressJs which interacts with Postman API, then perform CRUD Operations.

**Steps to implement the program 10**

**Step 1: Set Up Your Development Environment**

1. **Install Node.js:**

* Download and install Node.js from the [official website](https://nodejs.org/). This will also install npm, the Node package manager.

1. Create a folder in your system and open the created folder in Visual Studio Code.
2. **Create a Project Directory:**

* Open a terminal or command prompt and create a new directory for your project. Navigate into this directory.

mkdir my-express-app

cd my-express-app

1. **Initialize a New Node.js Project:**

* Initialize a new Node.js project with npm init. Follow the prompts to set up your package.json file.

npm init -y

**Step 2: Install Express.js**

1. **Install Express:**

* Install Express.js and any other necessary dependencies.

npm install express

**Step 3: Create the Express.js Application**

1. **Create the Application File:**

* Create a new file named app.js in your project directory.

1. **Write the Express.js Application Code:**

* Open app.js in your Visual Studio Code editor and add the following code:

const express = require('express');

const app = express();

const port = 3000;

app.use(express.json());

let items = []; // Sample in-memory data store

// Create - POST

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

const newItem = req.body;

items.push(newItem);

res.status(201).send(newItem);

});

// Read - GET

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

res.send(items);

});

app.get('/items/:id', (req, res) => {

const item = items.find(i => i.id === parseInt(req.params.id));

if (!item) return res.status(404).send('Item not found');

res.send(item);

});

// Update - PUT

app.put('/items/:id', (req, res) => {

const item = items.find(i => i.id === parseInt(req.params.id));

if (!item) return res.status(404).send('Item not found');

Object.assign(item, req.body);

res.send(item);

});

// Delete - DELETE

app.delete('/items/:id', (req, res) => {

const index = items.findIndex(i => i.id === parseInt(req.params.id));

if (index === -1) return res.status(404).send('Item not found');

const deletedItem = items.splice(index, 1);

res.send(deletedItem);

});

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

**Step 4: Run the Application**

1. **Start the Server:**

* In your terminal, run the following command to start your Express.js server.

node app.js

* You should see the message: Server running at http://localhost:3000.

**Step 5: Test the Application with Postman**

1. **Open Postman:**

* Open Postman to test your API endpoints.

1. **Create (POST) Operation:**

* Set the method to POST and enter the URL http://localhost:3000/items.
* Go to the Body tab, select raw, and choose JSON format.
* Enter JSON data, e.g.:

{

"id": 1,

"name": "Anil",

"description": "This is item Name"

}

* Click Send to create a new item.

1. **Read (GET) Operations:**
   * To get all items, set the method to GET and enter the URL http://localhost:3000/items, then click Send.
   * To get a single item by ID, set the method to GET and enter the URL http://localhost:3000/items/1, then click Send.
2. **Update (PUT) Operation:**
   * Set the method to PUT and enter the URL http://localhost:3000/items/1.
   * Go to the Body tab, select raw, and choose JSON format.
   * Enter the updated JSON data, e.g.:

{

"name": "Updated Item 1",

"description": "This is the updated item 1"

}

* + Click Send to update the item.

1. **Delete (DELETE) Operation:**
   * Set the method to DELETE and enter the URL http://localhost:3000/items/1.
   * Click Send to delete the item.





