IoT Device Management System

This project is a simple IoT (Internet of Things) device management system implemented using Node.js and Express for the server-side and HTML with JavaScript for the client-side.

Features

1. Device Registration:

- The `/register` endpoint allows users to register new devices by providing a unique device ID and specifying the device type.

- Validation ensures that both the device ID and device type are provided, and it checks for duplicate registrations.

2. Data Handling:

- The `/data` endpoint allows devices to send data to the server, updating the information associated with the specific device in 'devices.json'.

- The server logs the received data with timestamps.

3. Command Handling:

- The `/command` endpoint enables devices to send commands to the server, updating the commands associated with the specific device in 'devices.json'.

- Like data handling, the server logs the received commands with timestamps.

4. Device Display:

- The `/show` endpoint provides a list of all registered devices, allowing users to view the current state of registered devices.

5. Logging:

- Activities and events are logged to a 'logs.txt' file, providing a timestamped record of device registrations, data received, and commands received.

Server-Side (Node.js and Express)

Dependencies

- `express`: Web framework for Node.js.
- `fs`: File system module for reading and writing files.

Getting Started

1. Install dependencies: `npm install`
2. Start the server: `node server.js`

Endpoints

- **POST `/register`**: Register new devices.
- **GET `/show`**: Display all registered devices.
- **POST `/data`**: Receive data from devices.
- **POST `/command`**: Receive commands from devices.

Client-Side (HTML with JavaScript)

HTML Structure

- The HTML file contains forms for registering devices, sending data, and sending commands.

JavaScript Functions:

- `registerDevice()`, `sendData()`, and `sendCommand()` handle asynchronous communication with the server.
- `displayDevices()` fetches and updates the UI with the list of registered devices.

Usage

1. Open `index.html` in a web browser.
2. Use the provided forms to register new devices, send data, and send commands.
3. The UI will be updated in real-time to reflect the changes in device registrations and activities.

Notes and Considerations:

- Ensure that the server is running (`node server.js`) before interacting with the UI.

- The code currently uses a single 'devices. json' file to store both device information and commands. Consider separating this into two files for better organization.

- Handle potential security considerations, such as input validation and error handling improvements.

- Be aware of file path discrepancies between the code and the actual file locations.