Router Management Take-Home Assignment

Overview

This programming assignment is designed to evaluate your practical skills with HTTP, authentication, and automation using Node.js. It mirrors a real-world scenario where you need to interact with a network device (in this case, a router) programmatically by reverse engineering its web interface.

Objective

You will create a Node.js service that:

- Authenticates with a router web interface by inspecting network traffic.
- Retrieves device metadata via the router's API.
- Automates actions such as enabling/disabling Wi-Fi or firewall settings.

Target Device

You will be interacting with a router via its web interface. Use your browser's Developer Tools → Network tab to inspect:

- Login/authentication flow.
- API calls made when viewing or updating router settings.

Device URL:

Use the below device to perform the assignment.
Link: https://wifi-admin.netlify.app

Project Requirements

1. Router Interrogation

Create a REST API endpoint in Node.js that retrieves and returns metadata from the router including:

- Model

- Firmware Version
- MAC Address
- Serial Number
- Uptime
- Wi-Fi Status (enabled/disabled)
- Firewall Status
- Any other relevant information

NOTE: Some information might be missing, you can handle that by mentioning N/A

Authentication

You must identify how the router authenticates users (e.g., tokens, cookies, headers) and replicate this process in your code.

Router Interrogation Endpoint

- Method: POST
- Path: /api/router/interrogate

Example Request Body:

```
json
CopyEdit
{
  "username": "admin",
  "password": "admin"
}
```

Example Response:

```
json
CopyEdit
{
  "model": "XYZ123",
  "firmwareVersion": "v2.3.4",
  "macAddress": "00:11:22:33:44:55",
  "uptime": "5 days",
  "wifiStatus": "enabled",
  "firewallStatus": "disabled"
```

```
}
```

2. Router Actions

Implement REST API endpoints to control the router. At a minimum, include:

- POST /api/router/wifi/enable
- POST /api/router/wifi/disable
- POST /api/router/firewall/enable
- POST /api/router/firewall/disable

These endpoints must:

- Authenticate with the router
- Perform the correct action via the router's backend API
- Return success or failure responses with clear messages

Router Action Endpoint

- Method: POST
- Path: /api/router/action

Example Request Body:

```
json
CopyEdit
{
  "address": "192.168.0.1",
  "username": "admin",
  "password": "admin",
  "action": "enable_wifi"
}
```

You are free to structure the endpoints as needed as long as they are logical, consistent, and documented.

Bonus Points

- Provide a fallback unauthenticated version of the interrogation endpoint (if router allows limited info without login).
- Add support for basic router discovery on the local network (e.g., scanning common IP ranges).

Deliverables

- Node.js source code in a ZIP or github link
- `README.md` with setup instructions and API usage examples
- Proper error handling with error messages for different edge cases/error scenarios
- Well-structured and maintainable code

Evaluation Criteria

- Accurate API call reconstruction and automation
- Correct handling of authentication mechanisms
- Code quality, organization, and readability
- Robust error handling and edge-case coverage
- Documentation clarity and test coverage