LẬP TRÌNH WEB NÂNG CAO - 503106

BÀI TẬP THỰC HÀNH TUẦN 5

Giảng viên biên soạn: ThS. Mai Văn Mạnh

MỤC TIÊU BÀI THỰC HÀNH:

1. Ôn tập kiến thức về Rest API.
2. Gọi Rest API trong NodeJS bằng http và node-fetch module.
3. Tìm hiểu khái niệm Cross-Origin Resource Sharing (CORS).
4. Tìm hiểu khái niệm middleware và tự viết một middleware để đọc request body.
5. Xử lý dữ liệu trong HTML form bằng express-form, express-validator.
6. Sử dụng flash message để truyền dữ liệu giữa các routes.
7. Phòng chống tấn công DDOS bằng cơ chế rate limit (express-rate-limit).
8. Tiếp tục sử dụng các kiến thức ở các bài trước về nodejs, express… để phát triển ứng dụng web.

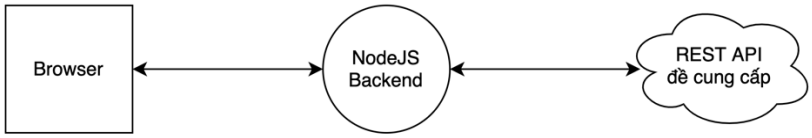
ĐỀ BÀI

Hãy sử dụng nodejs và các module như http, express, ejs,… để tạo một trang web hiển thị danh sách người dùng (với giao diện tương tự bài tập tuần trước). Khi click vào một người dùng thì trang web sẽ chuyển sang hiển thị thông tin chi tiết của người dùng đó. Ngoài ra, trang web cũng cung cấp các chức năng như: thêm một người dùng mới, cập nhật thông tin người dùng và xóa người dùng. Cần hiển thị modal dialog để xác nhận trước khi cập nhật hoặc xóa dữ liệu.

YÊU CẦU VỀ DỮ LIỆU:

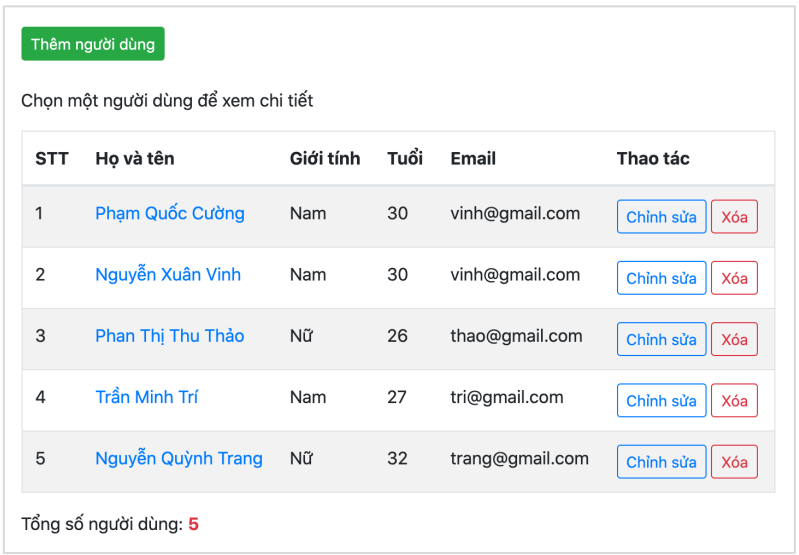
- Trang web (phía nodejs) **không** trực tiếp lưu dữ liệu mà sẽ sử dụng dữ liệu từ một Rest API.

- Rest API đã được cung cấp sẵn tại https://web-nang-cao.herokuapp.com/lab5/, sinh viên truy cập liên kết bên trên để xem thông tin mô tả về các api endpoints.
- Khi người dùng cần xem danh sách users, thêm user, cập nhật user hoặc xóa một user thì phía nodejs cần chuyển tiếp yêu cầu đến các api endpoints bên dưới để thực hiện chức năng tương ứng.
- Về mặt kỹ thuật, nodejs của chúng ta lúc này đóng vai trò như là một "forwarder", nó tiếp nhận yêu cầu từ người truy cập sau đó chuyển tiếp yêu cầu đến REST API, nhận lại kết quả rồi báo lại cho người truy cập trang web.



Ảnh minh họa các thành phần trong cơ chế hoạt động của ứng dụng web

Phía front end hoàn hoàn có thể sử dụng ajax/fetch để gọi trực tiếp đến rest api để lấy dữ liệu mà không cần thông qua phía nodejs backend. Tuy nhiên điều này chỉ có thể được thực hiện nếu cơ chế Cross-Origin Resource Sharing được thiết lập phù hợp. Mặc định ajax/fetch chỉ có thể đọc được dữ liệu ở phía server nếu client và server có cùng tên miền. Các ajax/fetch request từ front end đến một server khác tên miền sẽ bị block bởi hầu hết các trình duyệt vì lý do bảo mật. Trong khi đó phía NodeJS là back end, không liên quan gì đến các trình duyệt nên không bị kiểm soát bởi cơ chế này do đó có thể tự do đọc dữ liệu từ bất kỳ api nào.



Giao diện trang chủ minh họa

Các yêu cầu khác:

- Dữ liệu ở tất cả HTML form cần được kiểm tra ở phía server bằng cách sử dụng các middleware như express-form, express-validator…
- Trong các bài trước, body-parser được sử dụng để đọc dữ liệu từ request body. Trong bài tập này, sinh viên tự viết một body parser (middleware) của riêng mình, không sử dụng các module cài đặt từ bên ngoài.
- Sử dụng flash messsage để chuyển tiếp thông báo giữa các route trong express app.
- Sử dụng tính năng rate limit trong express để phòng chống tấn công DDOS.
- Nếu người dùng truy cập vào bất cứ path nào khác với mô tả ở trên thì chuyển hướng người dùng đến đường dẫn /error đồng thời hiển thị giao diện thông báo lỗi 404 not found.