**BÀI THỰC HÀNH 3**

**HOÀN THIỆN BACK-END CHO ỨNG DỤNG MINH HOẠ**

Bài 1: Thiết lập định tuyến cho các thao tác với review trong ứng dụng minh hoạ. Việc này cần thực hiện trong tệp tin movies.router.js

1.1 Định tuyến này sẽ có đường dẫn cuối cùng là ‘/review’

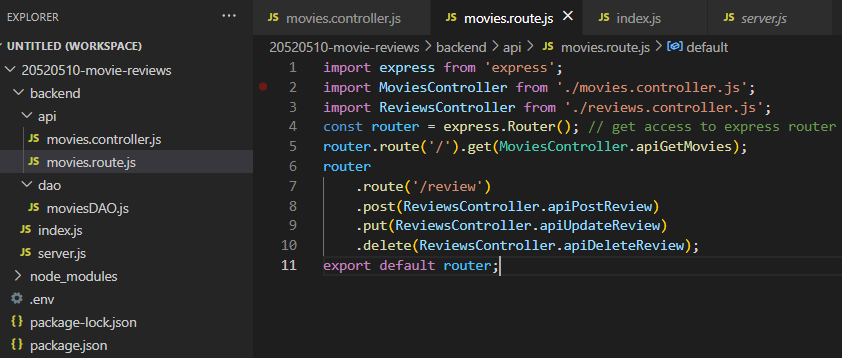
Ví dụ: localhost:3000/api/v1/movies/review

1.2 Thiết lập định tuyến thêm review vào db thông qua phương thức post, và gọi đến phương thức apiPostReview trong class ReviewsController trong tệp tin reviews.controller.js mà ta sẽ thiết lập ở bài 2.

1.3 Thiết lập định tuyến sửa review trên db thông qua phương thức put, và gọi đến phương

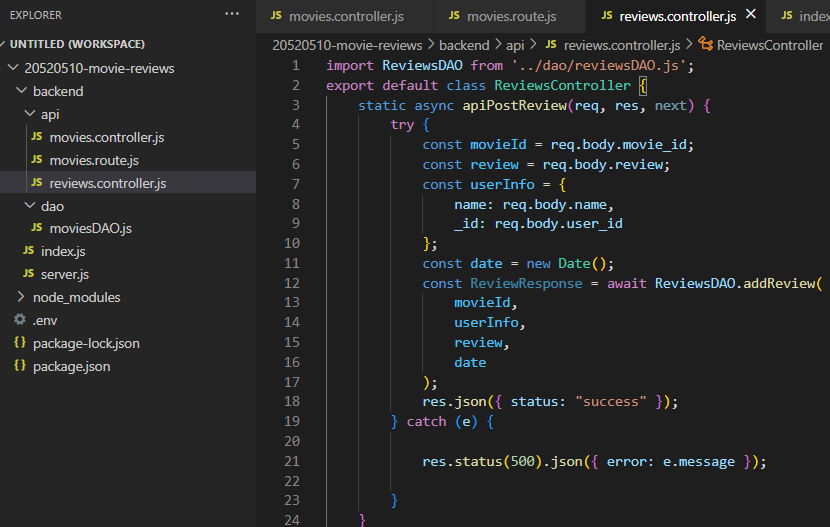
thức apiUpdateReview trong class ReviewsController trong tệp tin reviews.controller.js mà ta sẽ thiết lập ở bài 2.

1.4 Thiết lập định tuyến xoá review trên db thông qua phương thức delete, và gọi đến phương thức apiDeleteReview trong class ReviewsController trong tệp tin reviews.controller.js mà ta sẽ thiết lập ở bài 2.

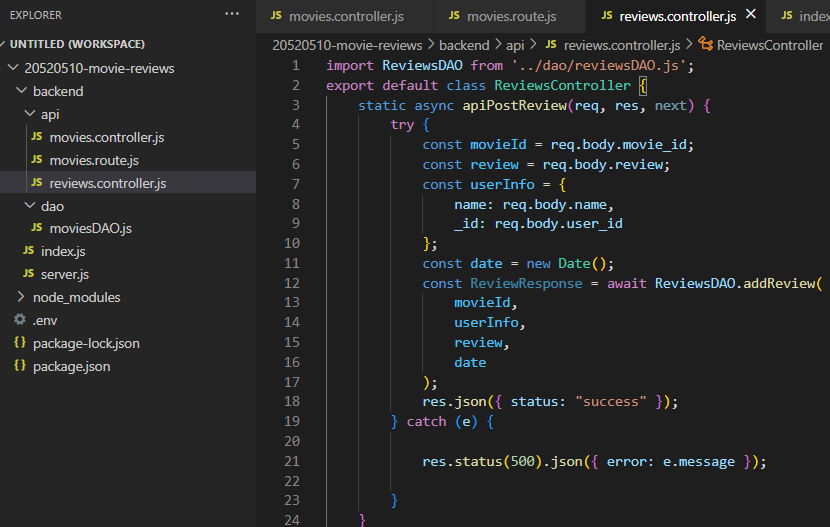


Bài 2: Thiết lập Controller cho review.

2.1 Tạo tệp tin reviews.controller.js trong thư mục api chứa một class có tên ReviewsController để quản lý các yêu cầu có liên quan đến review từ người dùng gửi lên từ máy khách.



2.2 Trong tệp tin vừa tạo ở bài 2.1 sẽ chứa dòng lệnh import nội dung từ tệp tin reviewsDAO.js (sẽ tạo ở bài 3) để gọi tới các hàm tương tác dữ liệu.



2.3 Tạo phương thức có tên apiPostReview() để quản lý các yêu cầu được gửi từ máy khách (thiết lập ở bài 1.2).

- Phương thức này sẽ lấy dữ liệu gửi lên từ người dùng thông qua tham số req gồm có

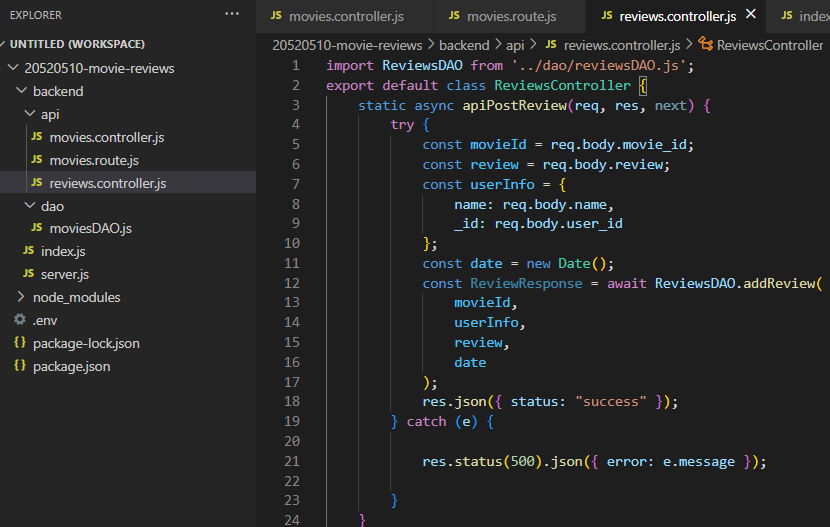
movie\_id, review, userinfo gồm name và id (gửi dưới dạng JSON trong body của req), và tạo ra một biến date để lưu trữ ngày tháng năm hiện tại của review.

- Sau đó, gọi đến hàm addReview() được định nghĩa trong ReviewsDAO (sẽ tạo ở bài 3)

để thêm được review vào db.

- Nếu thêm dữ liệu thành công sẽ trả về cho máy khách 1 thông báo dưới dạng JSON

báo ‘success’, nếu không thành công sẽ log lỗi ra trên màn hình console của terminal.



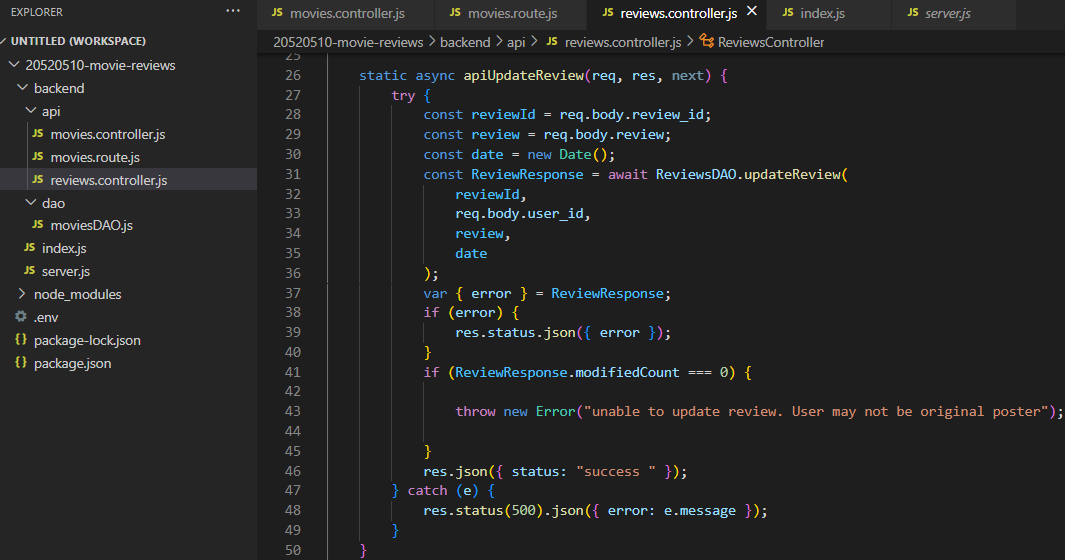
2.4 Tạo phương thức có tên apiUpdateReview() để quản lý các yêu cầu được gửi từ máy khách (thiết lập ở bài 1.3).

- Phương thức này sẽ lấy dữ liệu gửi lên từ người dùng thông qua tham số req gồm có review\_id, user\_id (gửi dưới dạng JSON trong body của req), và tạo ra một biến date để lưu trữ ngày tháng năm hiện tại của review mới, lưu ý, muốn update thành công review thì id user phải là user đã tạo ra review.

- Sau đó, gọi đến hàm updateReview() được định nghĩa trong ReviewsDAO (sẽ tạo ở bài 3) để sửa dữ liệu review trên db.

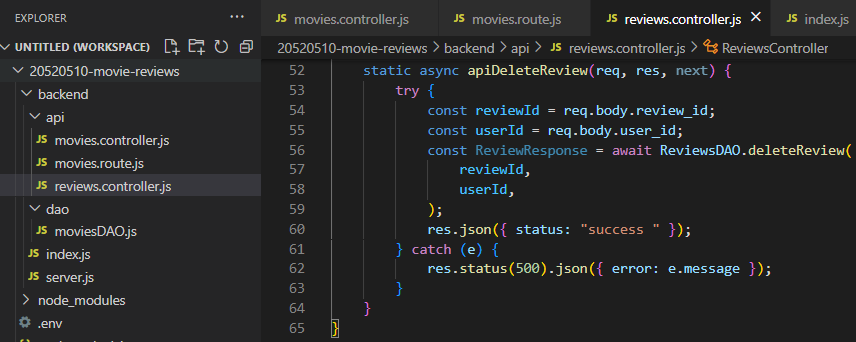
- Nếu thêm dữ liệu thành công sẽ trả về cho máy khách 1 thông báo dưới dạng JSON báo ‘success’, nếu không thành công sẽ log lỗi ra trên màn hình console của terminal.

- Lưu ý: Cần tạo ra biến ReviewResponse để lấy kết quả trả về khi gọi hàm updateReview(), vì hàm updateReview() trong bài 3 sẽ có trả về một biến tên là modifiedCount để xác định xem là có review nào thực sự đã được chỉnh sửa hay chưa.



2.5 Tạo phương thức có tên apiDeleteReview() để quản lý các yêu cầu được gửi từ máy khách (thiết lập ở bài 1.3).

- Phương thức này sẽ lấy dữ liệu gửi lên từ người dùng thông qua tham số req gồm có review\_id, user\_ id (gửi dưới dạng JSON trong body của req), và thực hiện việc xoá review thông qua hàm deleteReview trong lớp ReviewsDAO sẽ tạo ở bài 3.



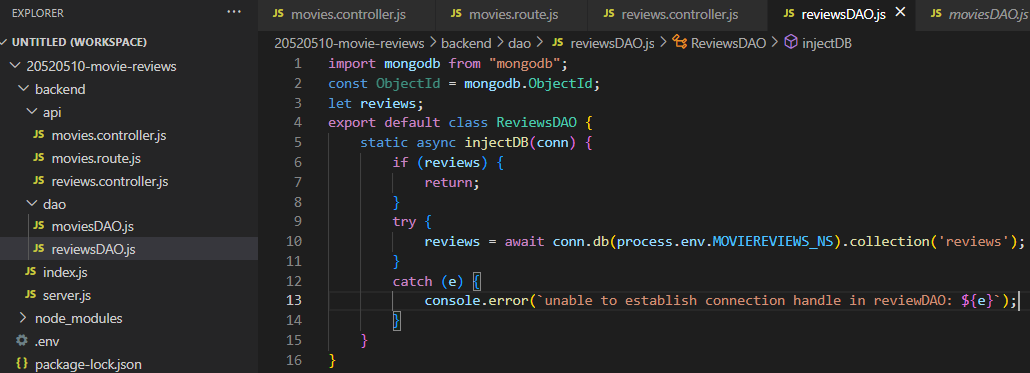
Bài 3: Thiết lập DAO cho reviews.

3.1 Trong thư mục DAO tạo tệp tin reviewsDAO.js.

- Tệp tin này cần import pakage mongodb đã cài đặt ở lab 2 để sử dụng một số phương thức cần thiết.

- Tạo một hằng số tên ObjectId = mongodb.ObjectId để sau này xử lý một số tác vụ liên quan đến trường dữ liệu \_id trong mongodb.

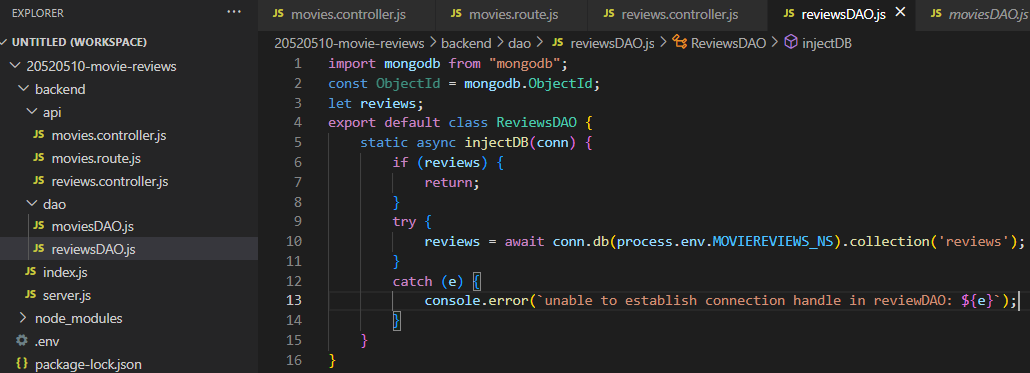
- Tạo ra một biến reviews để tham chiếu tới collection reviews sẽ tạo sau trên db.



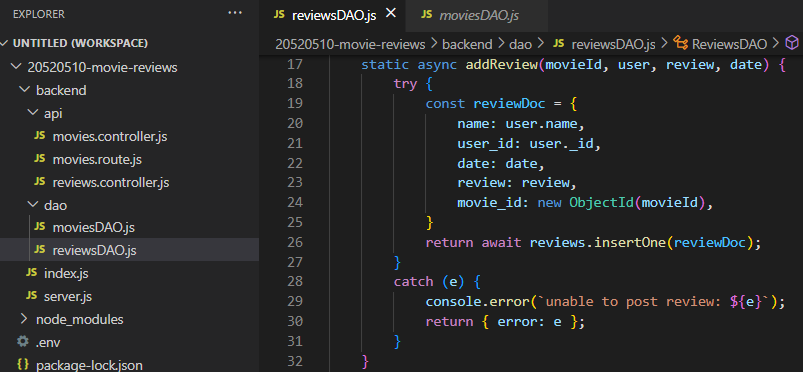
3.2 Tạo phương thức có tên injectDB() giúp kết nối tới collection tương ứng trên db.

- Cần tạo đối tượng và gọi phương thức injectDB() này trong tệp tin index.js để đảm bảo kết nối tới collection thành công, nhằm mục đích tương tác với dữ liệu.

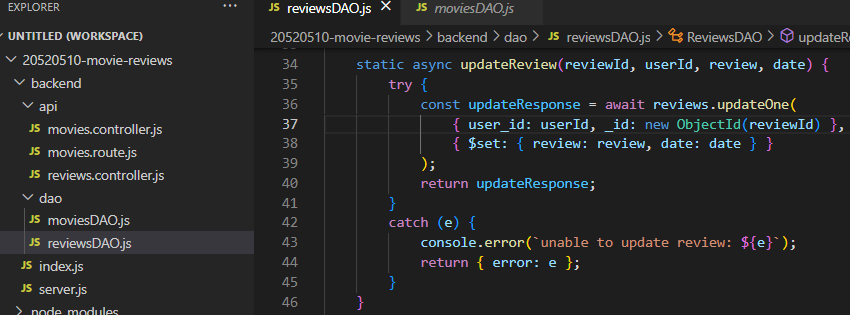
- Lưu ý: việc gọi injectDB() này trong tệp tin index phải sau dòng lệnh kết nối tới dữ liệu và trước khi khởi tạo máy chủ web.



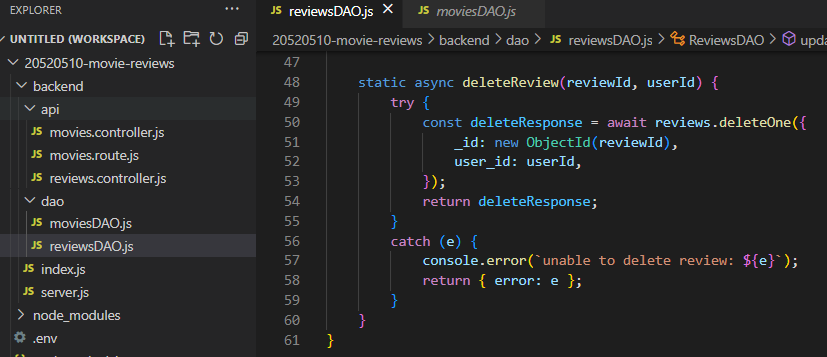
3.3 Tạo phương thức addReview() để thêm review vào db, trong hàm này sẽ có gọi một hàm insertOne(), lưu ý phải biến chuỗi movieId trong tham số truyền vào ở bài 2 thành dạng ObjectId.



3.4 Tạo phương thức updateReview() để sửa review trên db, trong hàm này sẽ có gọi một hàm updateOne(), lưu ý phải biến chuỗi reviewId trong tham số truyền vào ở bài 2 thành dạng ObjectId, phải cùng userId mới cho phép sửa review.



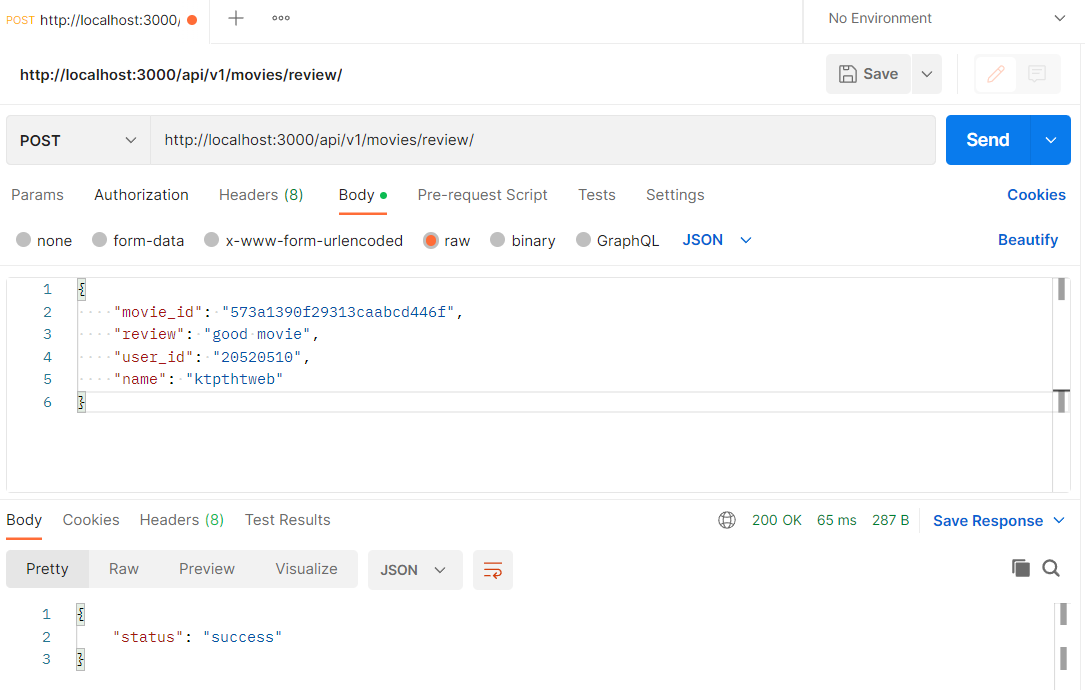
3.5 Tạo phương thức deleteReview() để thêm review vào db, trong hàm này sẽ có gọi một hàm deleteOne(), lưu ý phải biến chuỗi reviewId trong tham số truyền vào ở bài 2 thành dạng ObjectId, phải cùng userId mới cho phép xoá review.



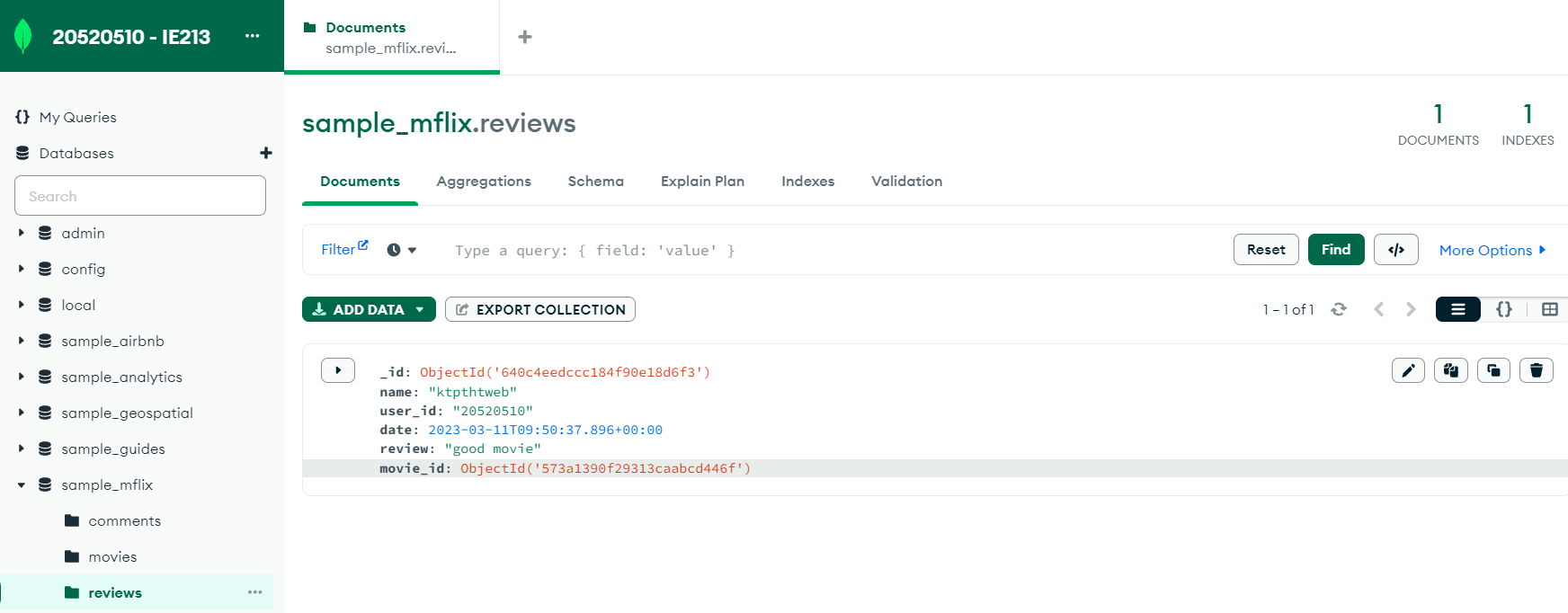
3.6 Thử nghiệm các API xem đã thành công hay chưa bằng phần mềm hỗ trợ như Insomnia: Yêu cầu đổi user\_id thành MSSV

Ví dụ: Thêm/Xoá/Sửa dữ liệu (tham khảo slide 60-63 trong bài học)

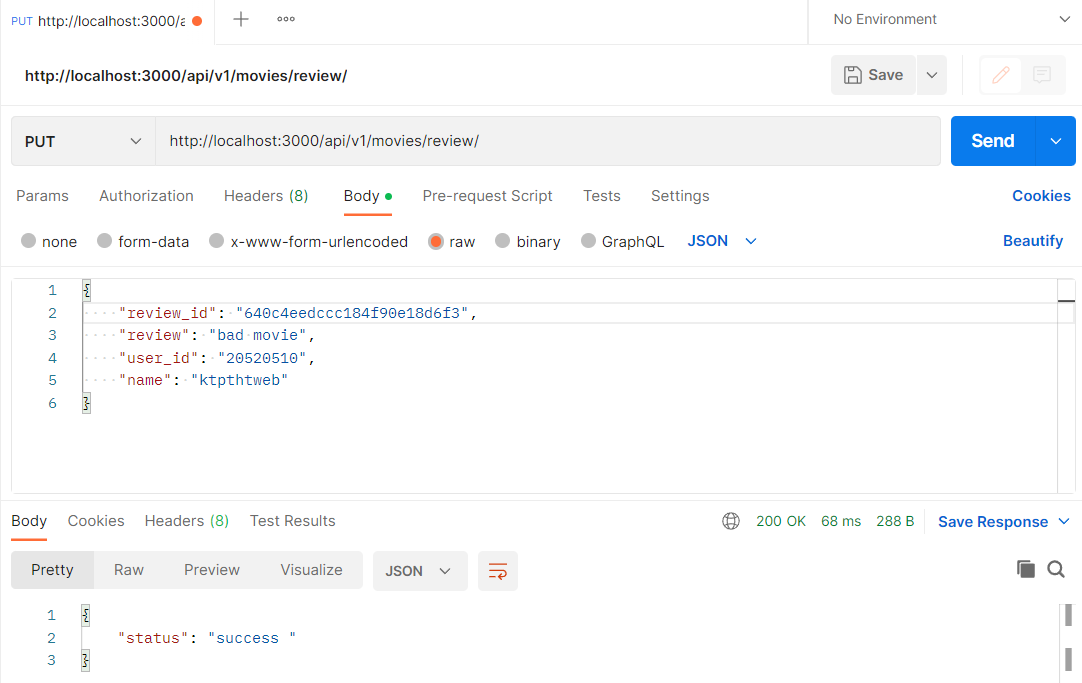
Thêm dữ liệu:



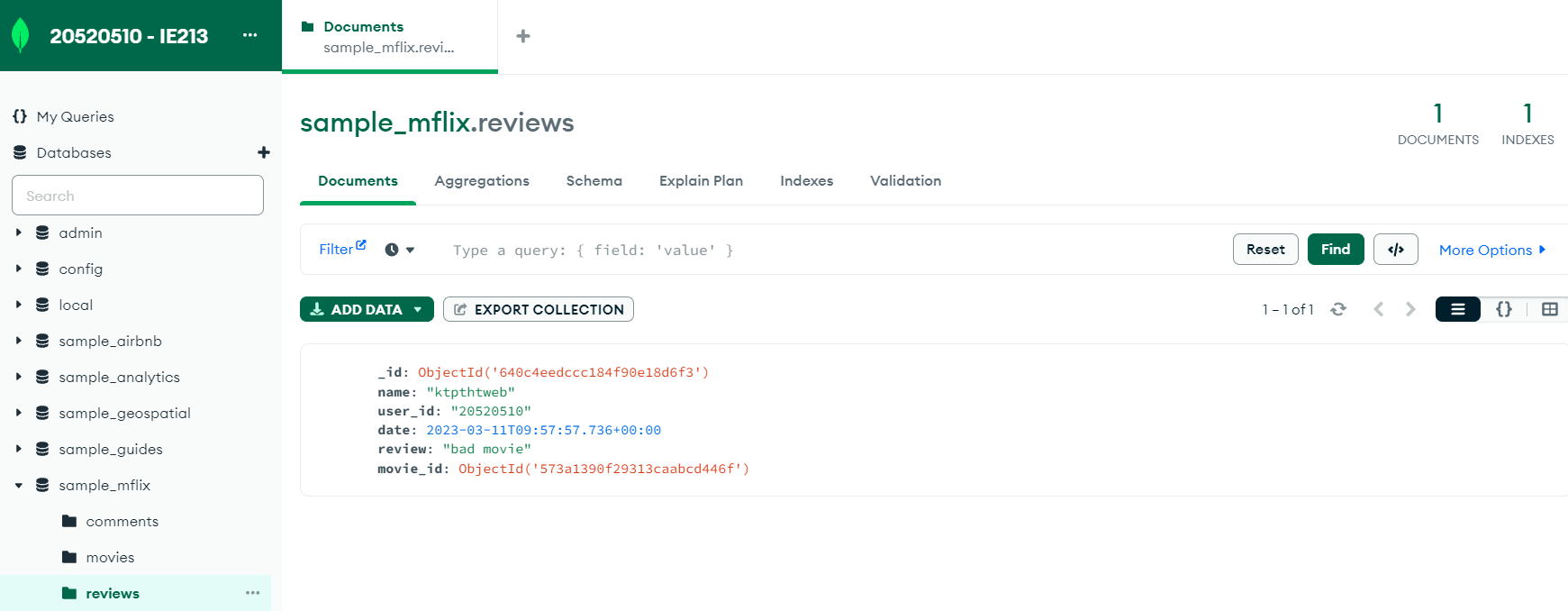
Kết quả sau khi thêm dữ liệu:



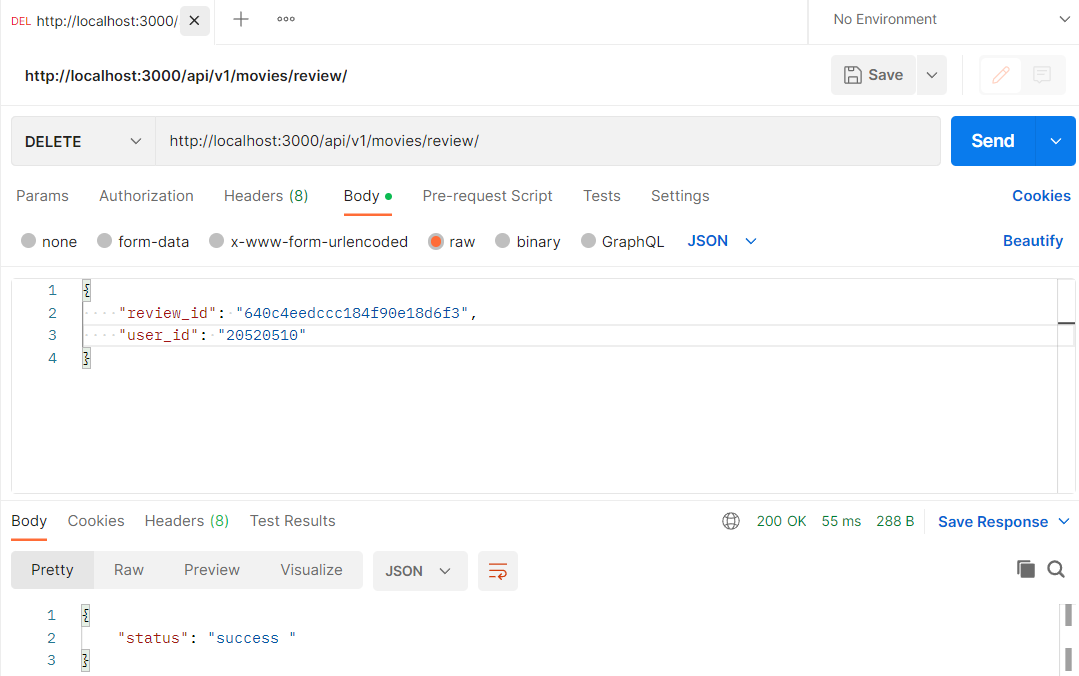
Sửa dữ liệu:



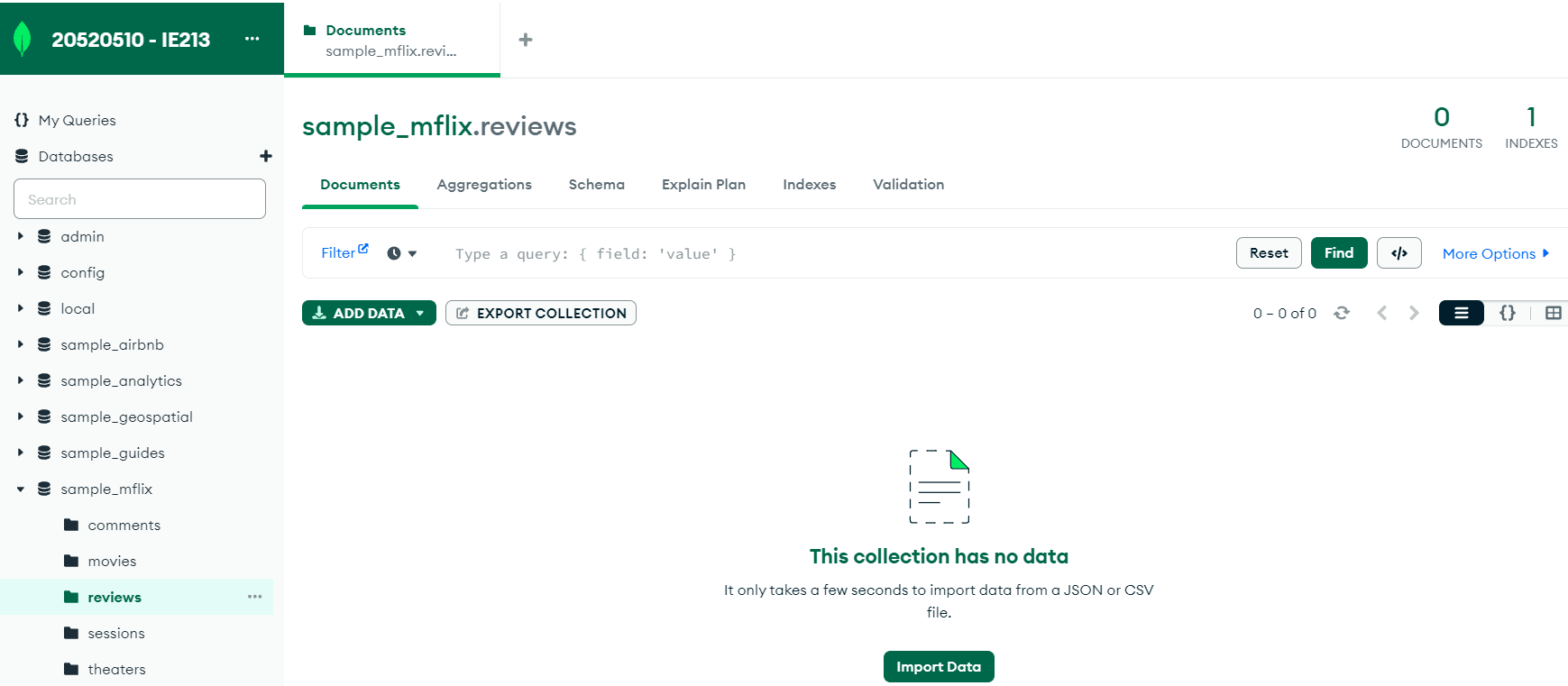
Kết quả sau khi sửa dữ liệu:



Xóa dữ liệu:



Kết quả sau khi xóa dữ liệu:

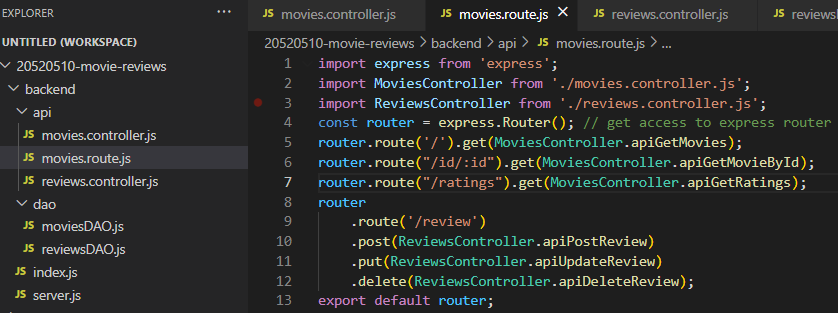


Bài 4: Hoàn thành back-end cho ứng dụng minh họa.

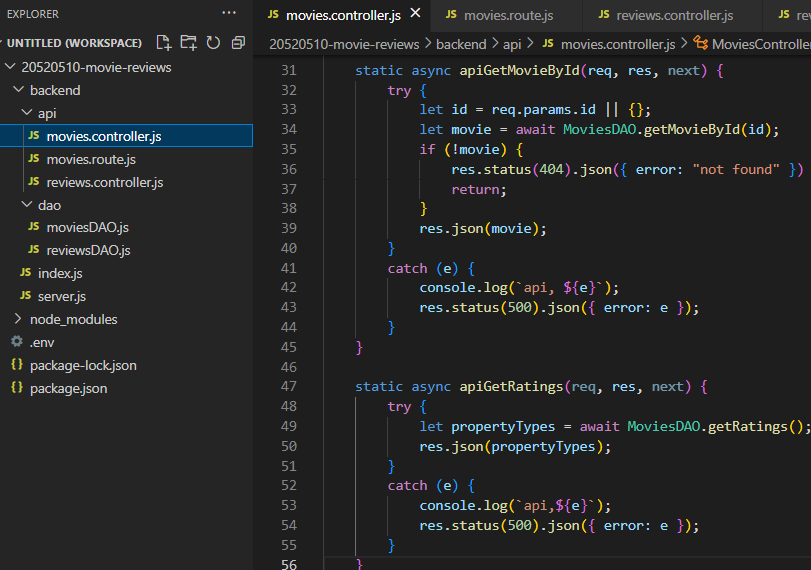
4.1 Thêm 2 định tuyến cho người dùng sử dụng các chức năng sau:

1. Lấy tất cả thông tin của phim và các review có liên quan dựa trên Id của phim.

2. Lấy tất cả các loại rating của phim trên dữ liệu.

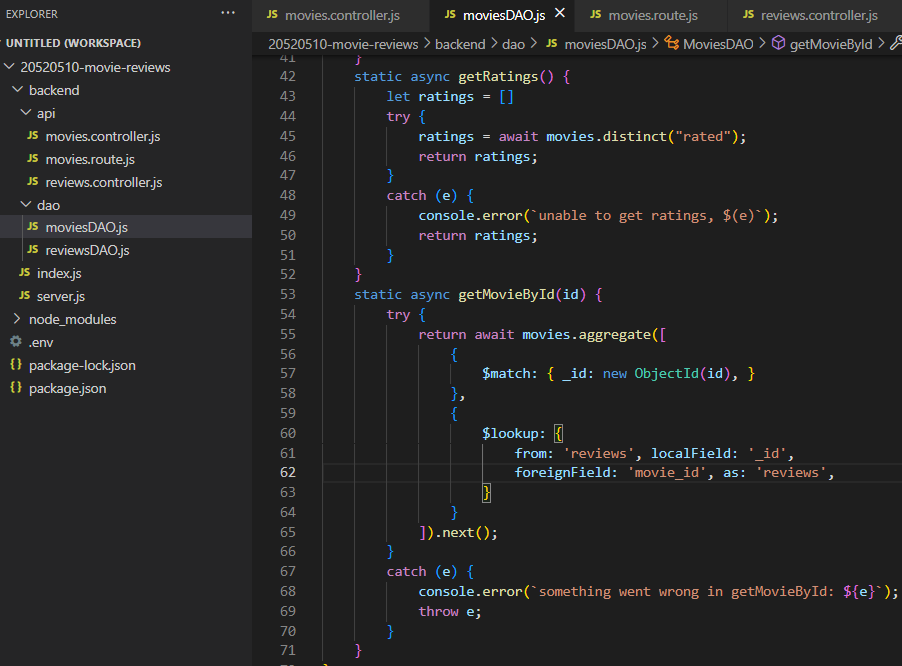


4.2 Thêm 2 phương thức controller tương ứng cho phần 4.1 là apiGetMovieById() và apiGetRatings() trong movie controller.



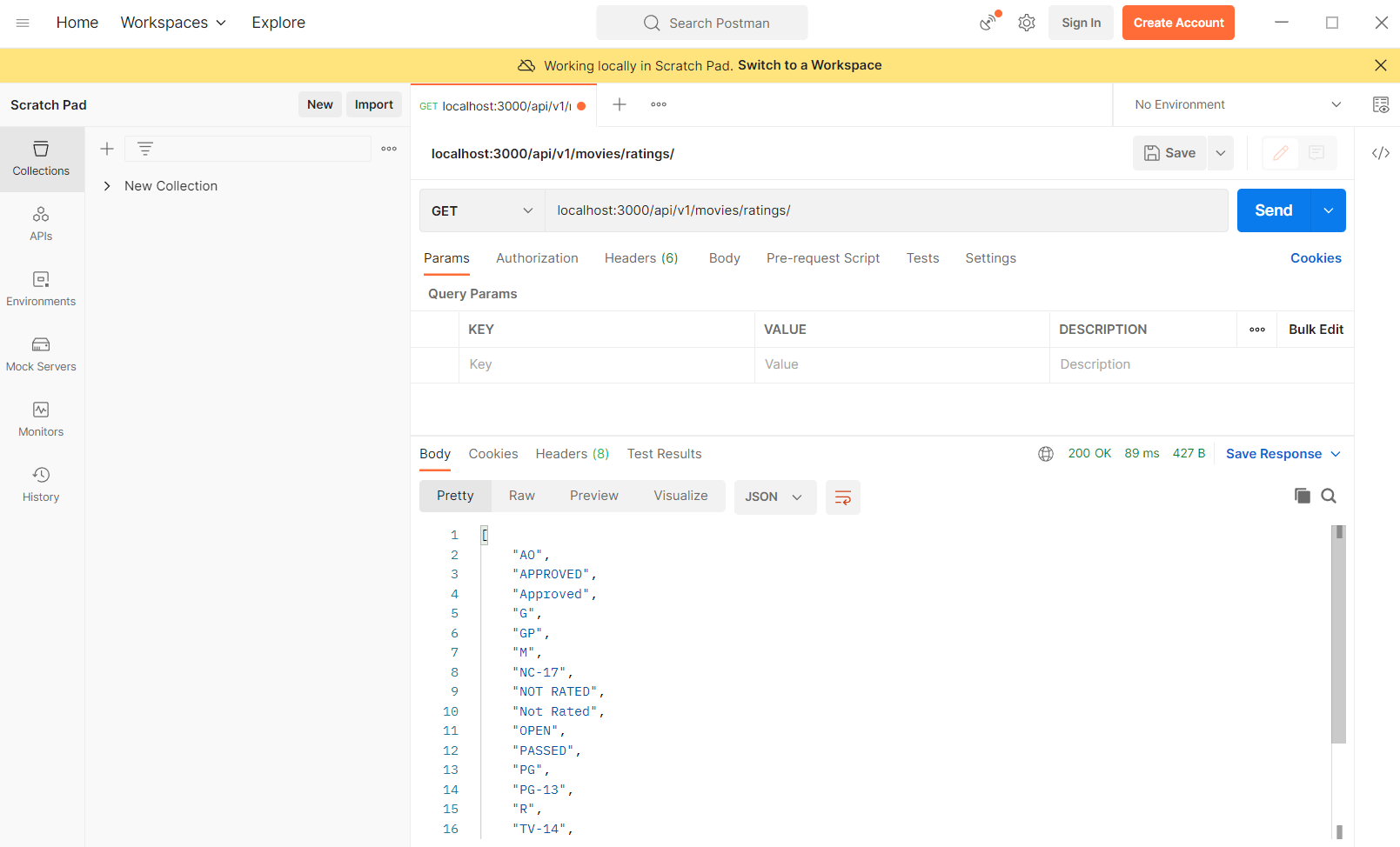
4.3 Thêm 2 phương thức DAO tương ứng cho phần 4.2 là getRatings() và getMovieById() trong dao movie.

Lưu ý: phần getMovieById() sẽ sử dụng một số toán tử và phương thức đã học ở bài thực hành 1 MongoDB như $match, $lookup (giống khóa ngoại trong SQL) và aggregate() để tổng hợp dữ liệu từ nhiều collection.



4.4 Thử nghiệm các API vừa tạo ở trên.

Thử nghiệm apiGetRatings() :



Thử nghiệm getMovieById():

