

Gage White, Gabrielle Akers, Jove Emmons, Nikki Gorski

Lobster Notes API documentation

## Professor Endpoints

### GET /api/professor/<int:prof\_id>/courses (Get Professor's Courses)

**Description:** Gets the list of all the courses taught by the logged in professor. Includes search functionality. This endpoint is consumed by the Professor Dashboard to populate the course list.

**Method:** GET

**Endpoint:** /api/professor/<prof\_id>/courses

**Path Params:** prof\_id (integer) – Unique ID of the professor

**Query Params:** search (string, optional) – Filters courses by partial match (Like %term or Subject LIKE %term%) on the course name/subject.

**Success Response:**

**Success Status (200 OK):** Returns a JSON array containing course details

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database connection or query execution failure.

### GET /api/course/<int:course\_id> (Get Single Course Details)

**Description:** Retrieves course information for a single course including professor's name. Endpoint is consumed by CourseDetail.jsx component to display course information and fetch the class roster.

**Method:** GET

**Endpoint:** /api/course/<course\_id>

**Path Params:** course\_id(integer) – Unique ID of course to retrieve

**Success Response:**

**Success Status (200 OK):** Returns single JSON object containing all course attributes plus the professor's name, retrieved through a LEFT JOIN on the user table.

**Error Responses:**

**404 Not Found:** Returns if the provided course\_id does not match an existing course.

**500 Internal Server Error:** Returns if there is a database connection or query execution failure.

**GET /api/course/<course\_id>/roster (Get Course Roster)**

**Description:** Retrieves list of students enrolled in a specific course. Endpoint is consumed by the Professor Dashboard to display the student roster for a selected course.

**Method: GET**

**Endpoint:** /api/course/<course\_id>/roster

**Path Params:** course\_id (integer) - The unique ID of the course whose roster is being requested

**Success Response:**

**Success Status (200 OK):** Returns a JSON array of student objects, including their basic user information and a list of enrolled courses.

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database query error while fetching the roster.

**POST /api/courses (Create a New Course)**

**Description:** Creates a new course record in the database, assigning it to the specified professor. Includes a database rollback for ACID if the insertion fails.

**Method: POST**

**ENDPOINT:** /api/courses

**Request Body:**

Field	Type	Required	Description
<b>Subject</b>	string	Yes	The academic subject code.
<b>CatalogNumber</b>	string	Yes	The course number.
<b>Name</b>	string	Yes	The full name of the course.
<b>Section</b>	int	Yes	The section number.
<b>Year</b>	int	Yes	The academic year.
<b>Session</b>	string	Yes	The academic session.
<b>ProfessorID</b>	int	Yes	The unique ID of the professor teaching the course.

**Success Response**

**Success Status (201 Created):** Returns the ID of the newly created course.

**Error Response**

**500 Internal Server Error:** Returns if there is a database failure or a required field is missing

**PUT /api/courses/{course\_id} (Update Course Details)**

**Description:** Allows updates of course details on an existing course by a logged in professor. Endpoint is consumed by Course Detail and enforced ACID via rollback if update fails.

**Method:** PUT

**Endpoint:** /api/courses/{course\_id}

**Path Params:** course\_id (integer) - The unique ID of the course to update.

**Request Body:**

Field	Type	Required	Description
Name	string	Yes	The new name of the course.
Section	int	Yes	The new section number.
Session	string	Yes	The new academic session.
Year	int	Yes	The new academic year.

**Success Response**

**Success Status (200 OK):** Success indication course was updated.

**Error Response**

**500 Internal Server Error:** Returns if there is a database failure, triggers rollback

**404 Not Found:** Returns if course\_id does not exist or if update was executed but made no changes.

**DELETE /api/courses/<course\_id> (Delete Course)**

**Description:** Permanently deletes a course identified by its unique ID. Will delete all records attached to course. Endpoint consumed by Course Detail component and enforces ACID with rollback if delete fails.

**Method: DELETE**

**Endpoint:** /api/courses/<course\_id>

**Path Params:** course\_id (integer) – unique ID of course to delete

**Success Response:**

**Success Status (200 OK):** Returns a message indication course was deleted.

### **Error Responses:**

**500 Internal Server Error:** Returns if there is a database failure, triggers rollback

**404 Not Found:** Returns if course\_id does not exist

## **User Endpoints:**

### **GET /api/users (List All Users)**

**Description:** Retrieves a list of all user accounts with simple details. Results are ordered by User\_ID

**Method:** GET

**Endpoint:** /api/users

**Success Response:**

**Success Status (200 OK):** Returns a JSON array containing basic info of all users

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database or query failure

## **Resources Endpoints:**

### **GET /api/resources (List and Search Resources)**

**Description:** Retrieves all resources stored in the system. Results are in descending order by creation date.

**Method:** GET

**Endpoint:** /api/resources

**Request Body:**

Query Params	Type	Optional	Description
search	string	Yes	<b>Search</b> across Topic, Keywords, Author, Body (for notes/PDFs), and resource Link/Url.
topic	string	Yes	Filters resources where the Topic field contains the search term.
format	string	Yes	Filters resources by a specific Format type. Search is case-insensitive.
subject	string	Yes	Filters resources where the Topic or Keywords contain the subject term.

**Success Response:**

**Success Status (200 OK):** Returns a JSON array of resources and/or URL where applicable

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database or query failure

**GET /api/resources/{resource\_id} (Get Single Resource Details)**

**Description:** Retrieves info on resources by ResourceID. Response includes format fields and average rating.

**Method:** GET

**Endpoint:** /api/resources/<resource\_id>

**Path Params:** resource\_id (int) – ID of resource to retrieve

**Success Response:**

**Success Status (200 OK):** Returns a JSON object with resource details

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database failure

**404 Not Found:** Returns if resource\_id does not exist

**POST /api/resources (Create a New Resource)**

**Description:** Create a new resource entry. Entire operation uses a single transaction and DB rollback implemented for ACID compliance if any part fails.

**Method: POST**

**Endpoint:** /api/resources

**Request Body:**

Field	Type	Required	Description
Author	string	Yes	The name of the user creating the resource.
Topic	string	Yes	The title or topic of the resource.
Format	string	Yes	The type of resource. Must be one of: Note, Website, Video, Pdf, or Image.
Body	string	No	The textual content for Notes or PDF bodies.
Link or Url	string	No	The external URL/address for non-Note formats (Website, Video, Pdf, Image).
DateFor	string	No	The date the resource is relevant to (YYYY-MM-DD); defaults to today.
Keywords	string	No	Comma-separated search terms or tags.
Duration	string	No	The duration for Video format resources.
Size	string	No	The file size for Image format resources.

**Success Response:**

**Success Status (201 Created):** Returns Id of created resource

### Error Responses:

**500 Internal Server Error:** Returns if there is a database failure. Will rollback transaction

**400 Bad Request:** Returns if required fields are missing

### POST /api/resources/{resource\_id}/ratings (Submit a Rating)

**Description:** Allows for user submitting rating for specific resource. Endpoint uses stored procedure SP\_Rating\_Rate and updates Rating column in resource table with new average score. Operation is ACID compliant as rollback occurs if process fails

**Method: POST**

**Endpoint:** /api/resources/<resource\_id>/ratings

**Path Params:** resource\_id (integer) – unique ID of resource being used

### Request Body:

Field	Type	Required	Description
Poster	string	Yes	The name of the user submitting the rating.
Score	float	Yes	The rating score. <b>Must be between 0.0 and 5.0.</b>
Date	string	Yes	The date the rating was submitted (YYYY-MM-DD).

### Success Response:

**Success Status (201 Created):** Returns a success message indicating rating was submitted and average rating was updated

### Error Responses:

**500 Internal Server Error:** Returns if there is a database failure, triggers rollback

**400 Bad Request:** Returns if fields are missing or if Score is outside range



### **GET /api/resources/{resource\_id}/ratings (Get Resource Ratings)**

**Description:** Retrieves ratings for specific resources. Results are ordered by descending date.

**Method:** GET

**Endpoint:** /api/resources/<resource\_id>/ratings

**Path Params:** resource\_id (integer) – unique ID of resource being used

**Success Response:**

**Success Status (200 OK):** Returns a JSON array of rating objects

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database or query failure

### **GET /api/subjects/<subject\_code>/resources (Get Resources by Subject)**

**Description:** Retrieves resources by subject code. Endpoint looks for subject name and returns a list of matching resources.

**Method** GET

**Endpoint:** /api/subjects/<subject\_code>/resources

**Path Params:** subject\_code (string) – code of subject

**Success Response:**

**Success Status (200 OK):** Returns a JSON object of subject and resources related to subject

**Error Responses:**

**500 Internal Server Error:** Returns if there is a database or query failure

**404 Not Found:** Returns if code does not match an existing subject in DB

