# Team 06

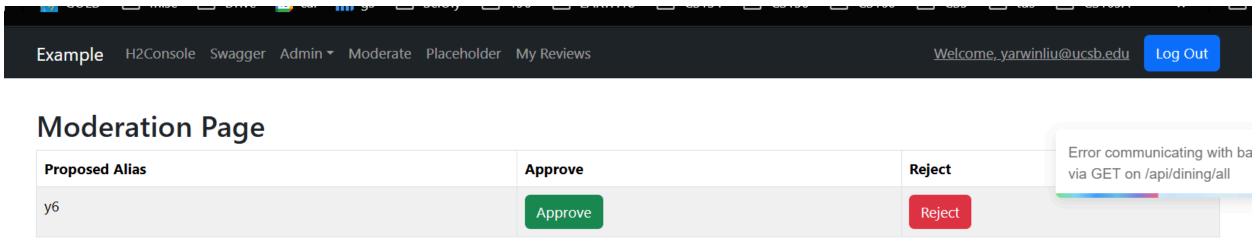
Sanjana Jilla, Riya Gupta, Yarwin Liu, Timothy Nguyen, Aryan Chopra, Saeed Arellano

**Link to repo:** https://github.com/ucsb-cs156-s25/proj-dining-s25-06

## User Facing Features

### 1. Moderator Table

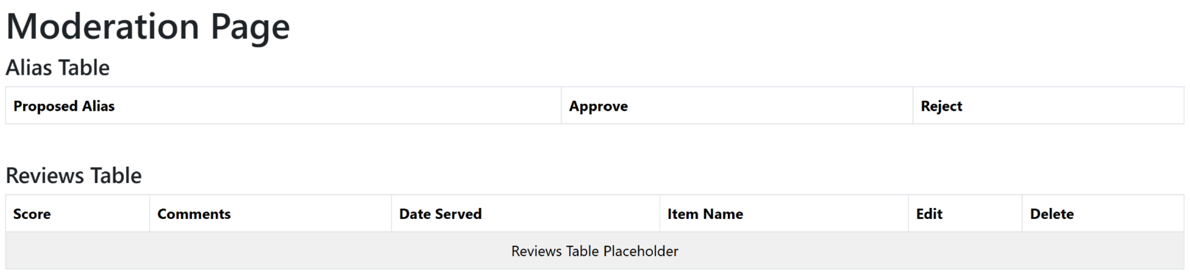
- Accessible by clicking the moderator tab on the navigation bar at the top of the screen
- Allows users with moderator role to approve or reject proposed aliases
- When the approve button is clicked, there is a pop up message confirming that the alias has been approved, and same with the reject
    - After the proposed alias is approved/rejected, it is removed from the table
- Additionally, there is a placeholder for the reviews table, as seen in the second screenshot



Moderation Page

| Proposed Alias | Approve | Reject |
|---|---|---|
| y6 | Approve | Reject |

Error communicating with ba
via GET on /api/dining/all



Moderation Page

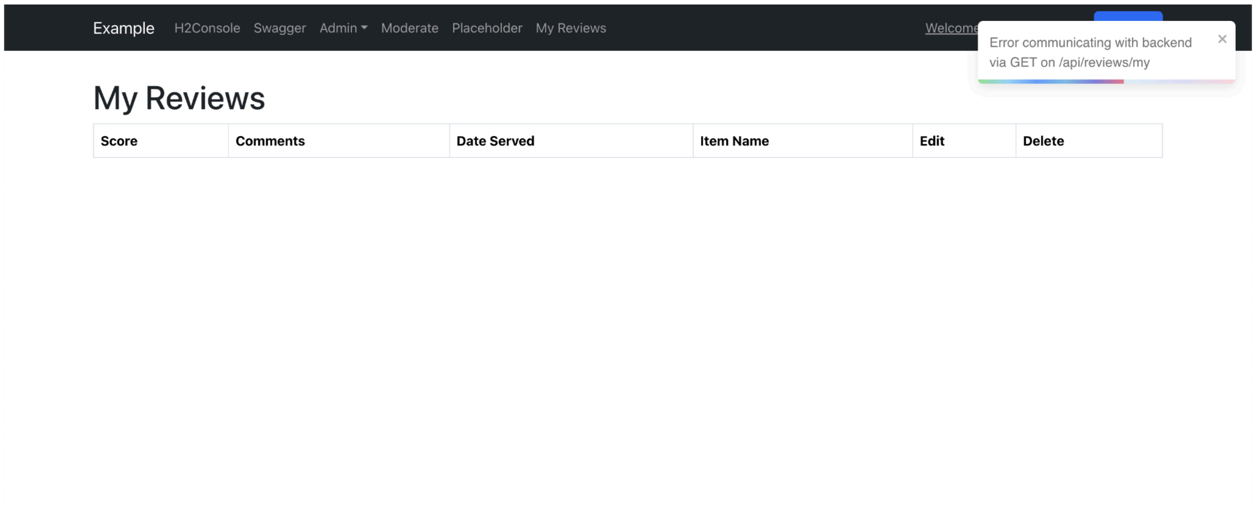
Alias Table

| Proposed Alias | Approve | Reject |
|---|---|---|

Reviews Table

| Score | Comments | Date Served | Item Name | Edit | Delete |
|---|---|---|---|---|---|
| Reviews Table Placeholder | | | | | |

## 2. MyReviews Page

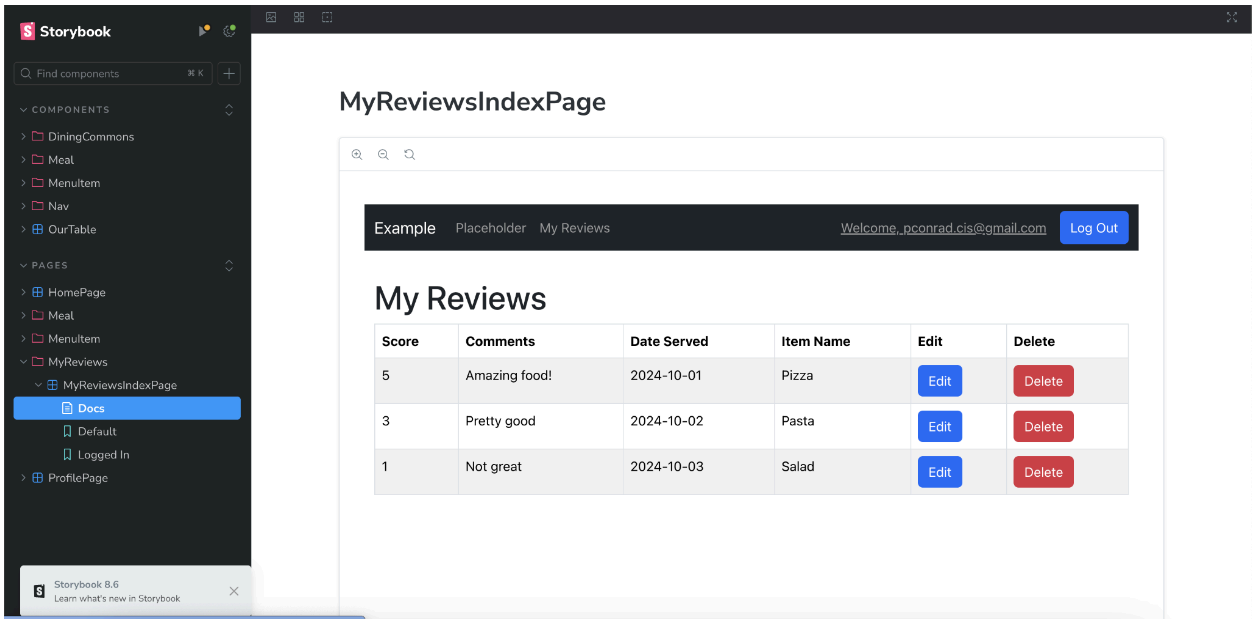

Example H2Console Swagger Admin Moderate Placeholder My Reviews Welcome

Error communicating with backend via GET on /api/reviews/my

My Reviews

| Score | Comments | Date Served | Item Name | Edit | Delete |
|---|---|---|---|---|---|

### Sample Storybook Entry:



MyReviewsIndexPage

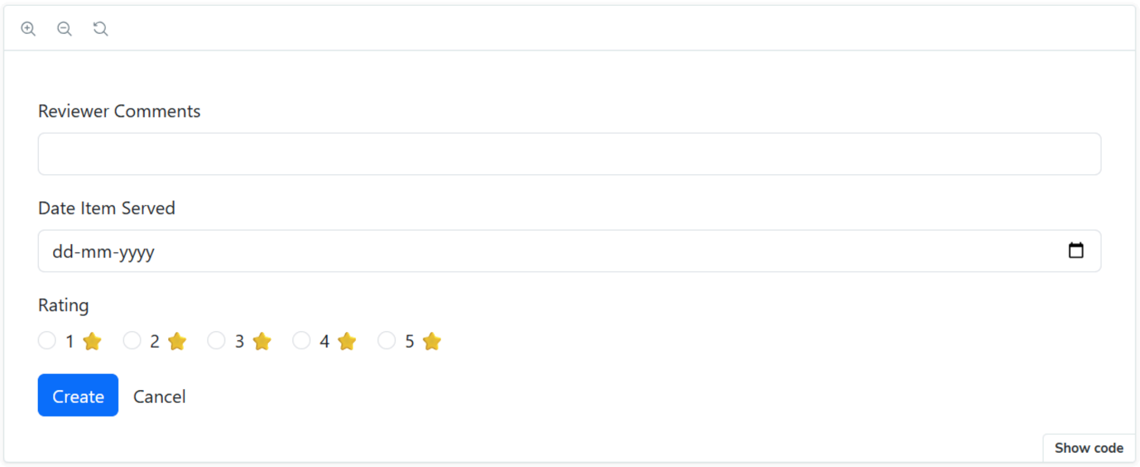
Example Placeholder My Reviews Welcome, pconrad.cis@gmail.com Log Out

My Reviews

| Score | Comments | Date Served | Item Name | Edit | Delete |
|---|---|---|---|---|---|
| 5 | Amazing food! | 2024-10-01 | Pizza | Edit | Delete |
| 3 | Pretty good | 2024-10-02 | Pasta | Edit | Delete |
| 1 | Not great | 2024-10-03 | Salad | Edit | Delete |

### User Interface for adding Reviews:

## ReviewForm



Reviewer Comments

Date Item Served

dd-mm-yyyy

Rating

1 ⭐ 2 ⭐ 3 ⭐ 4 ⭐ 5 ⭐

**Review Posting**

- Users can now view a table listing all reviews they have submitted by visiting the MyReviews page.

- The table includes key fields:
    - `id`: internal review identifier
    - `itemId`: the menu item being reviewed
    - `stars`: the rating (1–5)
    - `dateOfMeal`: when the meal was served
    - `comments`: optional text input by the user
    - `status`: moderation status (e.g., Awaiting Moderation)
    - `editedDate`: when the review was last modified
- The table supports edit and delete buttons for each review entry.

**Review Editing**

- When a user clicks “Edit” on a review, they are navigated to a pre-filled form where they can modify:
    - the number of stars
    - the text comments
- The user may then save the updated review, which sends a PUT request to the backend.

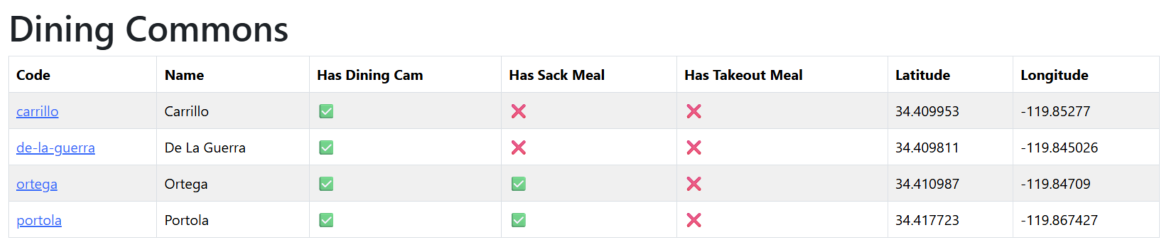
**Review Deletion**

- Clicking “Delete” on a review immediately removes the review after user confirmation.
- Deletion uses a DELETE request routed via the API and removes the entry from the UI upon success

## 3. Removing Longitude and Latitude

- The Dining Commons frontend fixtures used to load the latitude and longitude of each dining commons. This wasn’t really necessary so now it is taken out.

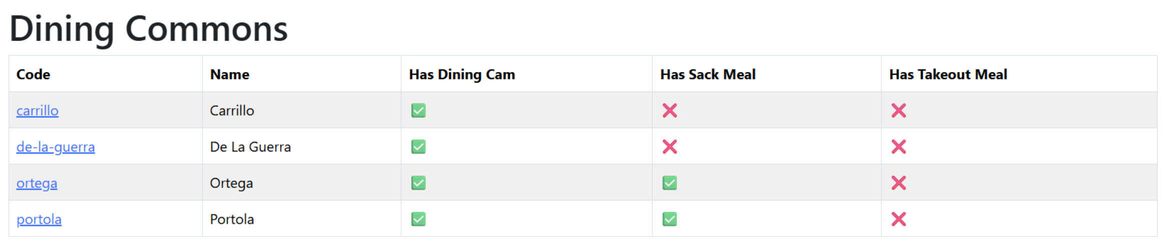
BEFORE:



### Dining Commons

| Code | Name | Has Dining Cam | Has Sack Meal | Has Takeout Meal | Latitude | Longitude |
|---|---|---|---|---|---|---|
| carrillo | Carrillo | ✅ | ❌ | ❌ | 34.409953 | -119.85277 |
| de-la-guerra | De La Guerra | ✅ | ❌ | ❌ | 34.409811 | -119.845026 |
| ortega | Ortega | ✅ | ✅ | ❌ | 34.410987 | -119.84709 |
| portola | Portola | ✅ | ✅ | ❌ | 34.417723 | -119.867427 |

AFTER:

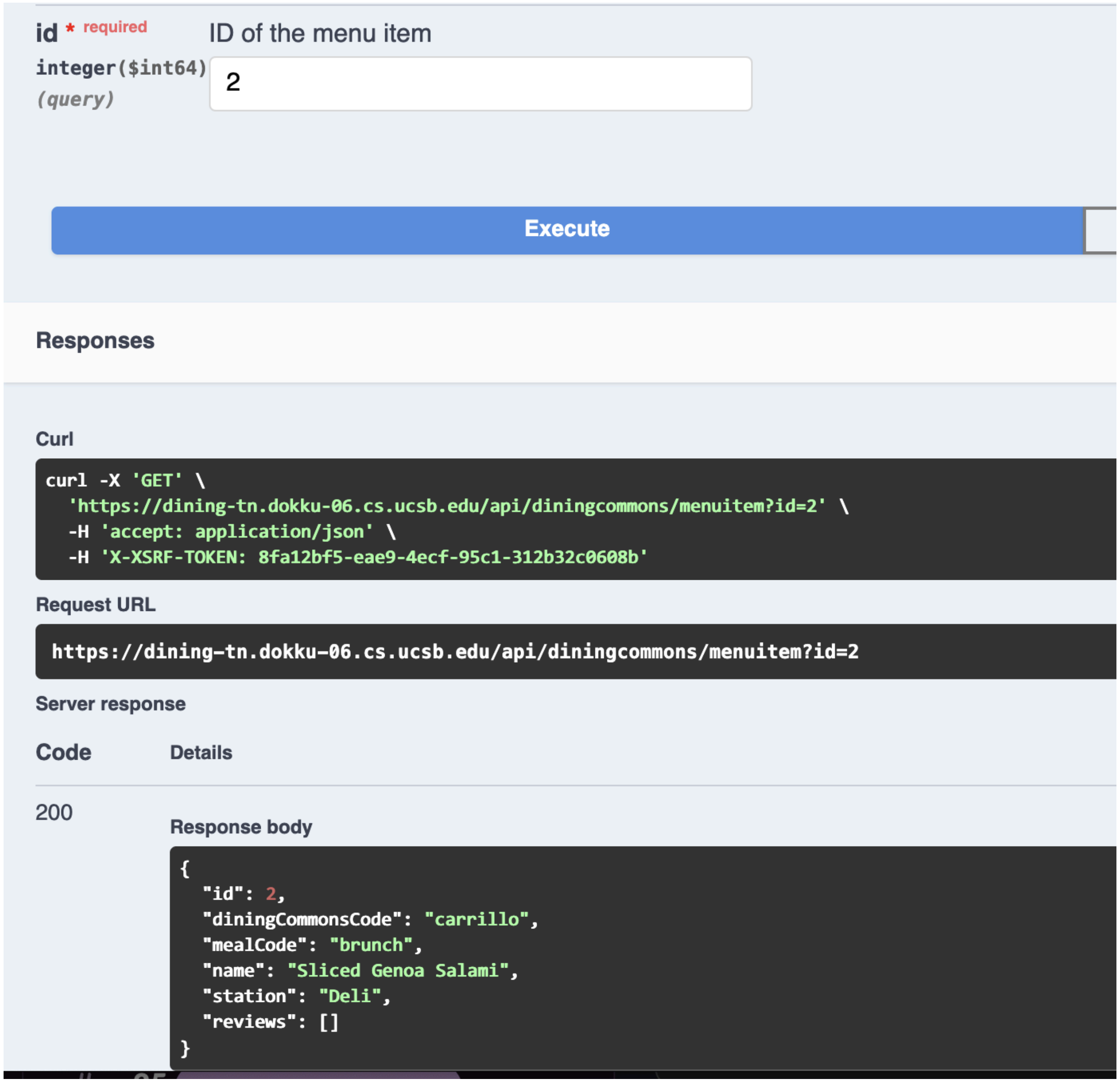


### Dining Commons

| Code | Name | Has Dining Cam | Has Sack Meal | Has Takeout Meal |
|---|---|---|---|---|
| carrillo | Carrillo | ✅ | ❌ | ❌ |
| de-la-guerra | De La Guerra | ✅ | ❌ | ❌ |
| ortega | Ortega | ✅ | ✅ | ❌ |
| portola | Portola | ✅ | ✅ | ❌ |

## 4. Individual Menu Item Get Endpoint

- Anyone who accesses the Swagger backend can look up one specific menu item by its Id.
- Looking up an ID that doesn't exist will throw a 404 response status



id * required
integer($int64)
(query)

ID of the menu item

2

Execute

**Responses**

Curl

```
curl -X 'GET' \
  'https://dining-tn.dokku-06.cs.ucsb.edu/api/diningcommons/menuitem?id=2' \
  -H 'accept: application/json' \
  -H 'X-XSRF-TOKEN: 8fa12bf5-eae9-4ecf-95c1-312b32c0608b'
```

Request URL

```
https://dining-tn.dokku-06.cs.ucsb.edu/api/diningcommons/menuitem?id=2
```

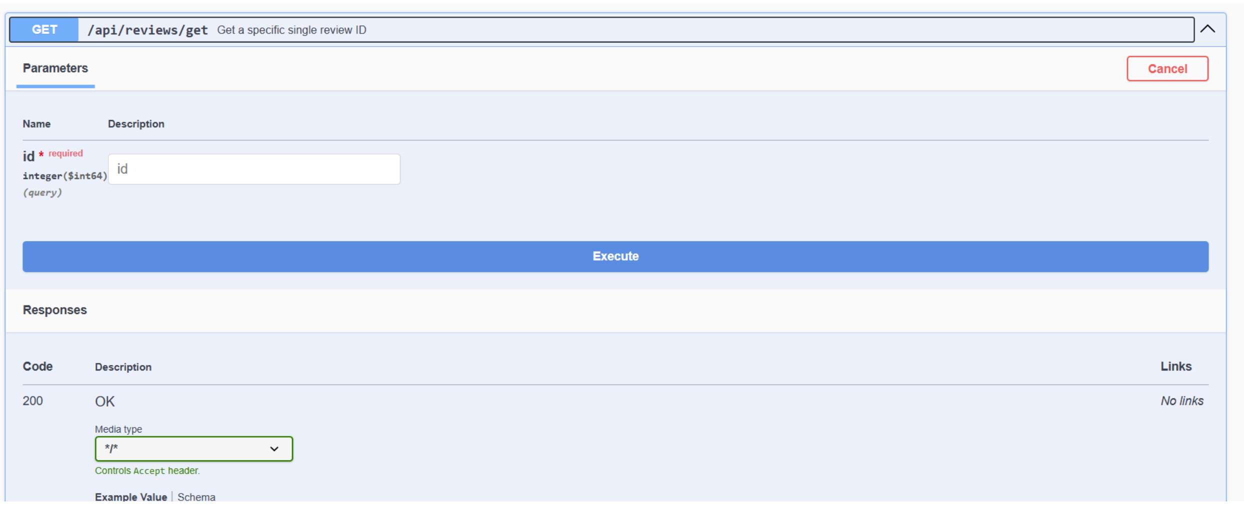
Server response

| Code | Details |
|---|---|
| 200 | Response body |

```
{
  "id": 2,
  "diningCommonsCode": "carrillo",
  "mealCode": "brunch",
  "name": "Sliced Genoa Salami",
  "station": "Deli",
  "reviews": []
}
```

## 5. Individual Reviews Get Endpoint

- Anyone who accesses the Swagger backend can look up one specific review by its Id. [MUST BE AN ADMIN OR USER WHO MADE THAT REVIEW]
- Looking up an ID that doesn't exist / ID that isn't yours will throw some sort of error


GET /api/reviews/get Get a specific single review ID

Parameters

Cancel

| Name | Description |
|---|---|
| id * required<br>integer($int64)<br>(query) | id |

Execute

Responses

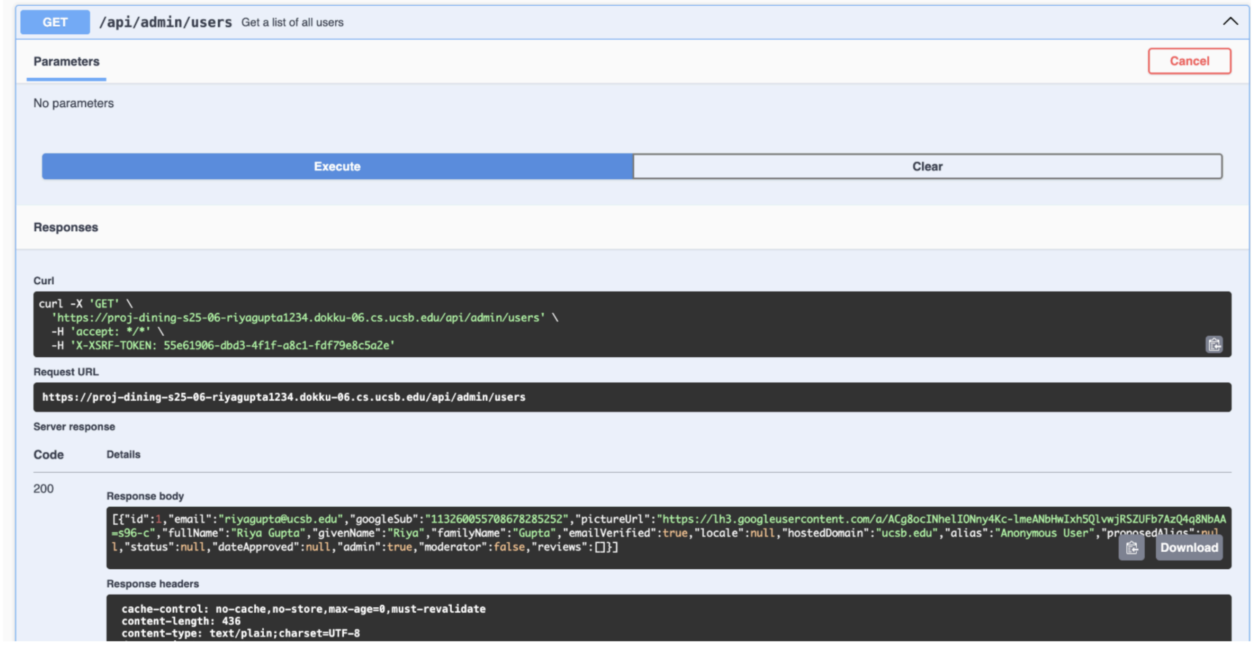
| Code | Description | Links |
|---|---|---|
| 200 | OK<br>Media type<br>*/*<br>Controls Accept header.<br>Example Value \| Schema | No links |

# Developer Features

**Summary of implementations for Developers:**

## 1. Added Moderator Role + Access To Endpoints

- Added a moderator role and gave it access to the necessary endpoints.
- On Swagger, visible as a field when we get all users.



```
curl -X 'GET' \
  'https://proj-dining-s25-06-riyagupta1234.dokku-06.cs.ucsb.edu/api/admin/users' \
  -H 'accept: */*' \
  -H 'X-XSRF-TOKEN: 55e61906-dbd3-4f1f-a8c1-fdf79e8c5a2e'
```

Request URL

```
https://proj-dining-s25-06-riyagupta1234.dokku-06.cs.ucsb.edu/api/admin/users
```

Server response

Code Details

200

Response body

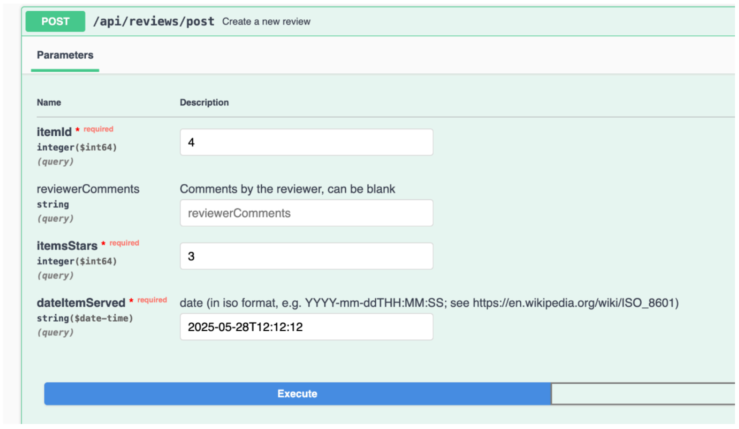
```
[{"id":1,"email":"riyagupta@ucsb.edu","googleSub":"113260055708678285252","pictureUrl":"https://lh3.googleusercontent.com/a/ACg8ocINhelIONny4Kc-lmeANbHwIxh5QlvwjRSZUFb7AzQ4q8NbAA
=s96-c","fullName":"Riya Gupta","givenName":"Riya","familyName":"Gupta","emailVerified":true,"locale":null,"hostedDomain":"ucsb.edu","alias":"Anonymous User","proposedAlias":nul
l,"status":null,"dateApproved":null,"admin":true,"moderator":false,"reviews":[]}]
```

Response headers

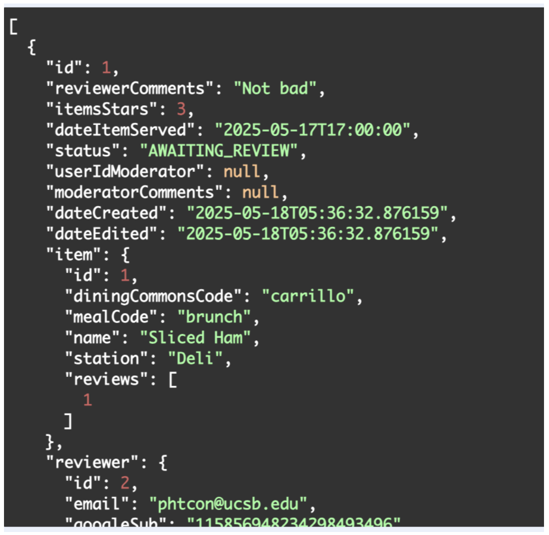
```
cache-control: no-cache,no-store,max-age=0,must-revalidate
content-length: 436
content-type: text/plain;charset=UTF-8
```

## 2. Auto Approve Reviews for Reviewers Who Put No Comments

- Changed Swagger endpoints to allow reviews that have no comments to automatically go through without a moderator review
- When we get post/edit a review without comments, it no longer shows up in "needs moderation"



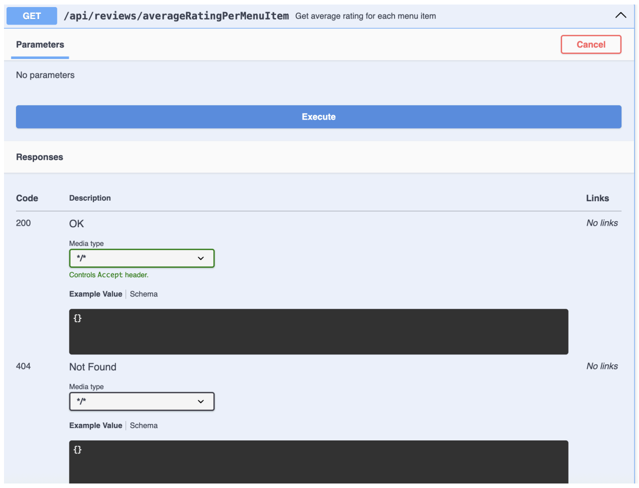
| Name | Description |
|---|---|
| itemId * required integer($int64) (query) | 4 |
| reviewerComments string (query) | Comments by the reviewer, can be blank<br>reviewerComments |
| itemsStars * required integer($int64) (query) | 3 |
| dateItemServed * required string($date-time) (query) | date (in iso format, e.g. YYYY-mm-ddTHH:MM:SS; see https://en.wikipedia.org/wiki/ISO_8601)<br>2025-05-28T12:12:12 |



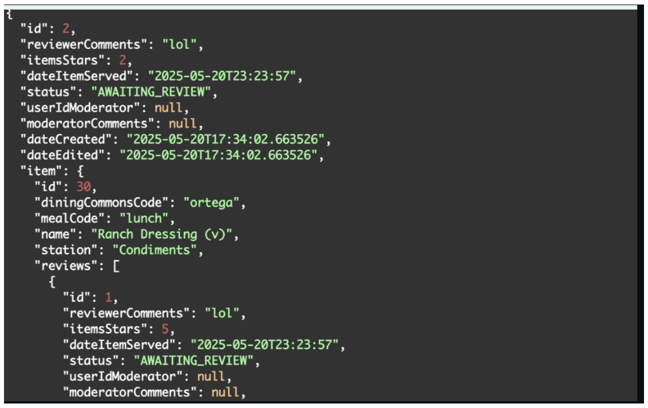
```
[
  {
    "id": 1,
    "reviewerComments": "Not bad",
    "itemsStars": 3,
    "dateItemServed": "2025-05-17T17:00:00",
    "status": "AWAITING_REVIEW",
    "userIdModerator": null,
    "moderatorComments": null,
    "dateCreated": "2025-05-18T05:36:32.876159",
    "dateEdited": "2025-05-18T05:36:32.876159",
    "item": {
      "id": 1,
      "diningCommonsCode": "carrillo",
      "mealCode": "brunch",
      "name": "Sliced Ham",
      "station": "Deli",
      "reviews": [
        1
      ]
    },
    "reviewer": {
      "id": 2,
      "email": "phtcon@ucsb.edu",
      "googleSub": "115856948234298493496"
```

3. AverageRatingPerMenuItem with .get endpoint

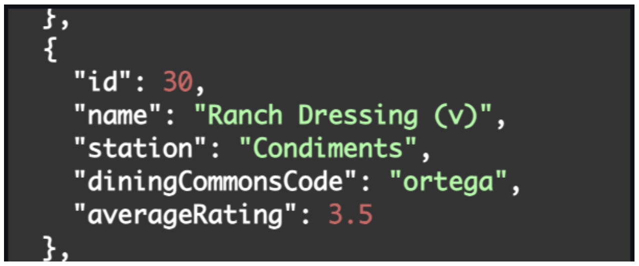
- Added a .get() endpoint to get all the the average rating per menu item when a review is entered.
- Utilized by Reviews Controller.



Example of the backend endpoint:



```
{
  "id": 2,
  "reviewerComments": "lol",
  "itemsStars": 2,
  "dateItemServed": "2025-05-20T23:23:57",
  "status": "AWAITING_REVIEW",
  "userIdModerator": null,
  "moderatorComments": null,
  "dateCreated": "2025-05-20T17:34:02.663526",
  "dateEdited": "2025-05-20T17:34:02.663526",
  "item": {
    "id": 30,
    "diningCommonsCode": "ortega",
    "mealCode": "lunch",
    "name": "Ranch Dressing (v)",
    "station": "Condiments",
    "reviews": [
      {
        "id": 1,
        "reviewerComments": "lol",
        "itemsStars": 5,
        "dateItemServed": "2025-05-20T23:23:57",
        "status": "AWAITING_REVIEW",
        "userIdModerator": null,
        "moderatorComments": null,
```



```
  },
  {
    "id": 30,
    "name": "Ranch Dressing (v)",
    "station": "Condiments",
    "diningCommonsCode": "ortega",
    "averageRating": 3.5
  },
```

4. Updated React-Router

- Updated React-Router 7 to upgrade the packages
- Changed all the react-router-dom imports to react router to fully implement the update

## PR Summary:

| Issue | s25-06 |
|---|---|
| 27 FEATURE: Adding a Moderator Role to the Backend | PR39 PR48 |
| 28 FEATURE: Adding a Moderator Role to the Frontend | - |
| 29 FEATURE: Update to React-Router 7 | PR47 PR49 |
| 30 BUG / CLEAN-UP: Remove the Latitude and Longitude Columns | PR32 PR33 |
| 31 FEATURE: Average Score | PR45 |
| 33 FEATURE: Create a Review Table | PR29 PR43 |
| 34 FEATURE: Implement the My Reviews Page | - |
| 35 CLEAN UP / SMALL FEATURE: Auto-Approve reviews without comments | PR37 |
| 36 SMALL FEATURE: Create a Placeholder Page for Posting a Review | PR30 |
| 37 FEATURE: Create a Form for Posting a Review | PR46 |
| 38 SMALL FEATURE: Add an Individual Menu Item Endpoint | PR34 PR35 PR36 |
| 39 FEATURE: Implementing the Post a Review Page | - |
| 40 SMALL FEATURE: Add a Reviews Placeholder Page | - |
| 41 FEATURE: Implement the Individual Reviews Page | - |
| 42 SMALL FEATURE: Create an Edit Placeholder Page | PR31 |
| 44 SMALL FEATURE: Add an Endpoint for an Individual Review | PR36 |
| 46 FEATURE: Add an Alias Approval Table | PR40 |
| 47 FEATURE: Implement the Moderation Page | PR43 |
| 48 FEATURE: Repeat the Deploy step in Github Actions to avoid failures | PR38 |