

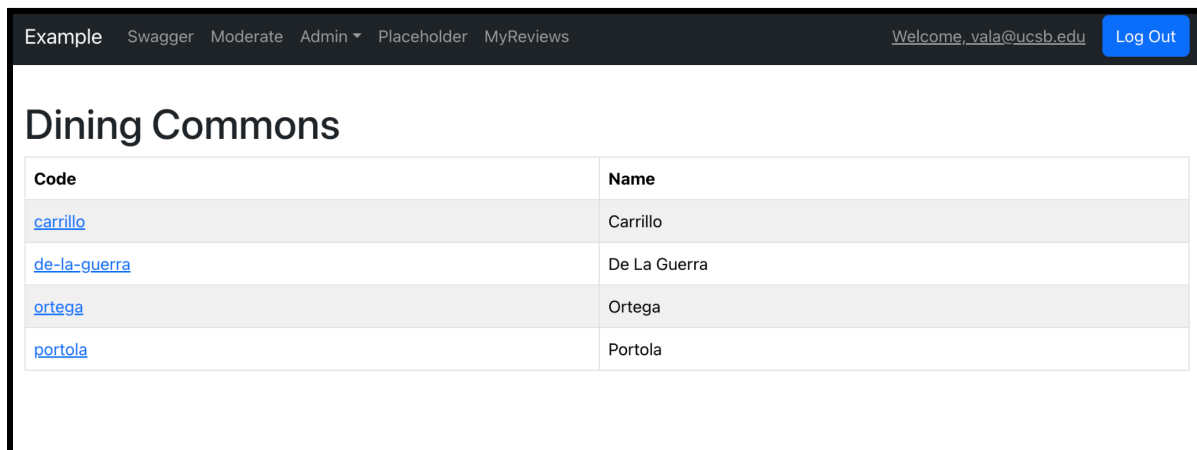
## Release Notes for Dining Commons Reviews Application

Vala Bahrami, Oviya Seeniraj, Aman Desai, Kevin Yan, Riona Pampati, Jennifer Zhu

### For Users:

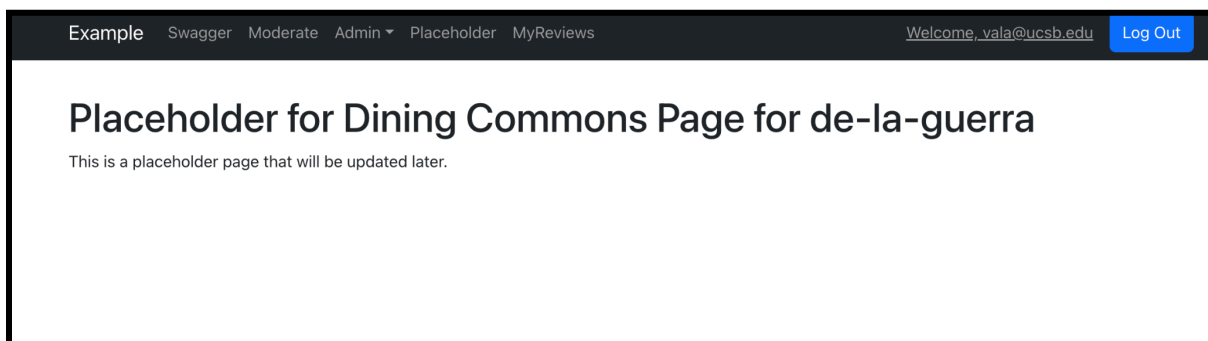
#### New Features:

- Home Page with Dining Commons List:
  - View a list of all UCSB dining commons, including Carrillo, De La Guerra, Ortega, and Portola.
  - Each dining commons name links to its dedicated page for more details.



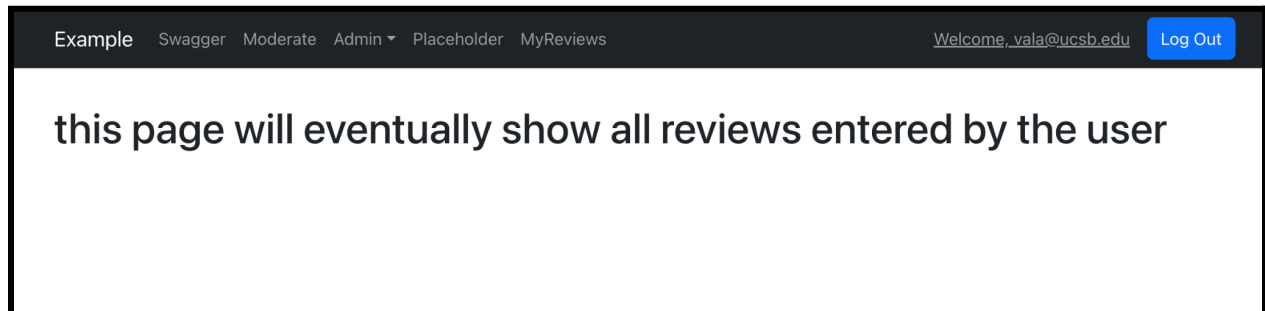
Code	Name
<a href="#">carrillo</a>	Carrillo
<a href="#">de-la-guerra</a>	De La Guerra
<a href="#">ortega</a>	Ortega
<a href="#">portola</a>	Portola

- Dining Commons Pages (placeholder):
  - The frontend page for each individual diningcommons is currently a placeholder, and users can not interact with it beyond visiting it.
  - When implemented and integrated with the backend, users will be able to view meals offered on that day at that dining commons, such as breakfast, lunch, and dinner.

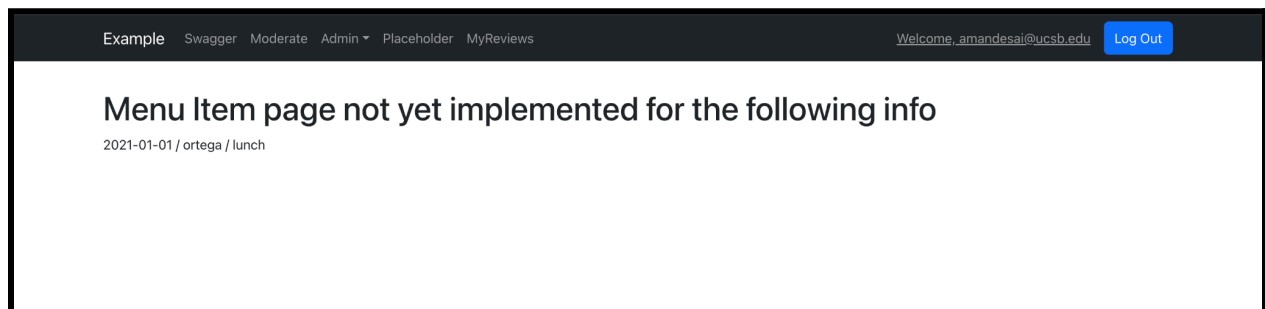


Placeholder for Dining Commons Page for de-la-guerra
This is a placeholder page that will be updated later.

- My Reviews Page (placeholder):
  - Access your personal "My Reviews" page from the navigation bar.
  - This frontend page is currently a placeholder, when implemented, users will be able to see all their submitted reviews here.



- Menu Item Page (placeholder):
  - Access a specific menu item's page by specifying the date-time, dining commons, and meal. Dynamically renders the page based on the URL.
  - This frontend page is currently a placeholder, once it is implemented, users will be able to see all reviews for this specific menu item.



### Improvements Over Previous Version:

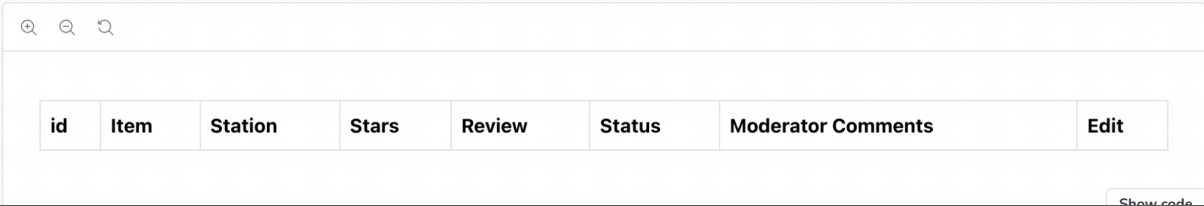
- This is effectively the first version of this application, but the previous version we built upon was a generic example app. Therefore, these improvements reflect changes from the example app.
- Streamlined Navigation:
  - Clean and intuitive user interface for easy navigation between pages.
  - Removed unnecessary pages related to UCSB Dates and Restaurants for a focused experience.
- Enhanced Data Accuracy:
  - Real-time data fetched from the UCSB Dining API ensures up-to-date menus.
- Responsive Design:
  - The application is optimized for both desktop and mobile devices.

## **For Developers:**

### **Backend Enhancements:**

- Table Component for user reviews:
  - Implemented a table to display a users's reviews in a table with the ability to edit them using a button.
  - On the moderators side, the table also includes a button to Approve or Reject the review.

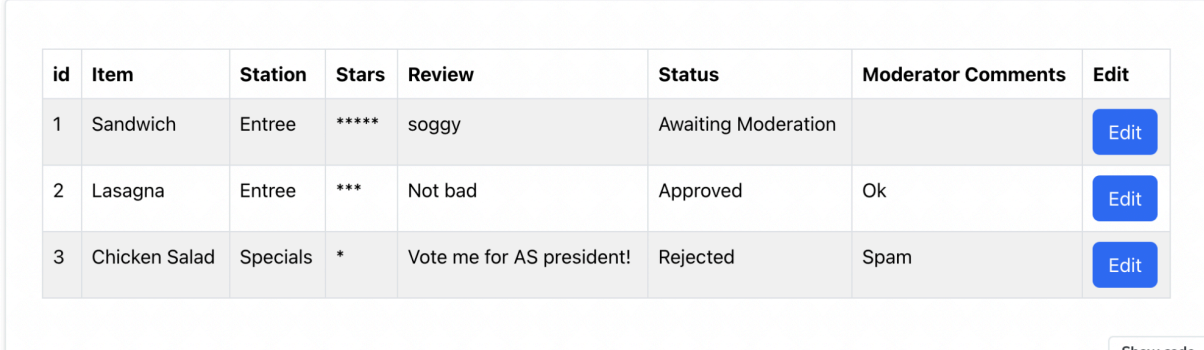
### MyReviewsTable



id	Item	Station	Stars	Review	Status	Moderator Comments	Edit
----	------	---------	-------	--------	--------	--------------------	------

Show code

### Three Items Ordinary User



id	Item	Station	Stars	Review	Status	Moderator Comments	Edit
1	Sandwich	Entree	*****	soggy	Awaiting Moderation		Edit
2	Lasagna	Entree	***	Not bad	Approved	Ok	Edit
3	Chicken Salad	Specials	*	Vote me for AS president!	Rejected	Spam	Edit

Show code

## Three Items Moderator

id	Item	Station	Stars	Review	Status	Moderator Comments	Edit	Approve	Reject
1	Sandwich	Entree	*****	soggy	Awaiting Moderation		Edit	Approve	Reject
2	Lasagna	Entree	***	Not bad	Approved	Ok	Edit	Approve	Reject
3	Chicken Salad	Specials	*	Vote me for AS president!	Rejected	Spam	Edit	Approve	Reject

[Show code](#)

## Three Items User Delete

id	Item	Station	Stars	Review	Status	Moderator Comments	Edit	Delete
1	Sandwich	Entree	*****	soggy	Awaiting Moderation		Edit	Delete
2	Lasagna	Entree	***	Not bad	Approved	Ok	Edit	Delete
3	Chicken Salad	Specials	*	Vote me for AS president!	Rejected	Spam	Edit	Delete

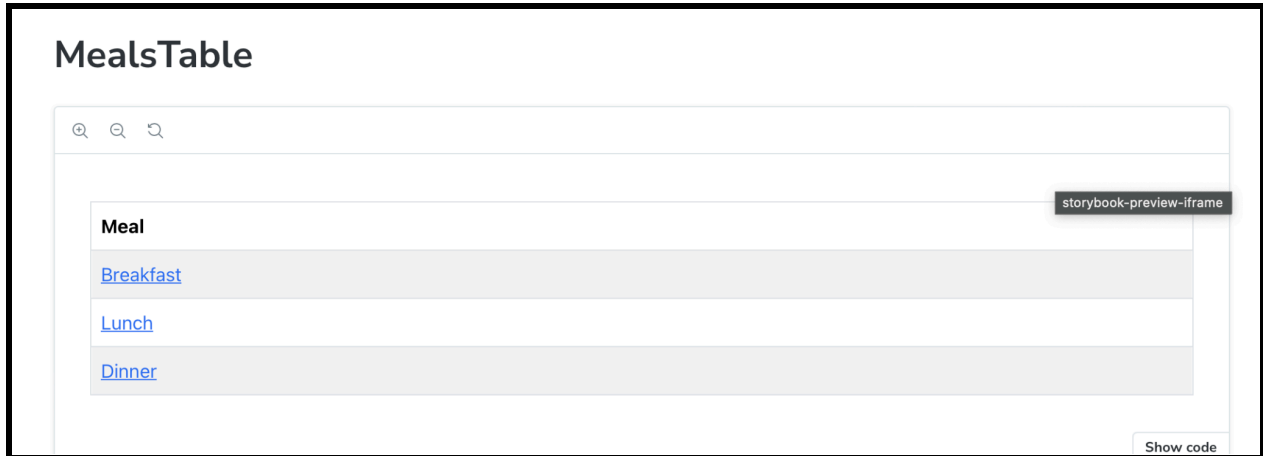
[Show code](#)

## Three Items Moderator Delete

id	Item	Station	Stars	Review	Status	Moderator Comments	Edit	Delete	Approve	Reject
1	Sandwich	Entree	*****	soggy	Awaiting Moderation		Edit	Delete	Approve	Reject
2	Lasagna	Entree	***	Not bad	Approved	Ok	Edit	Delete	Approve	Reject
3	Chicken Salad	Specials	*	Vote me for AS president!	Rejected	Spam	Edit	Delete	Approve	Reject

[Show code](#)

- Meals Table:
  - Created a table that holds the different meals for that dining commons.



- Database Setup:
    - Implemented new database migrations for `Review` and `MenuItem` entities.
    - Reviews database includes fields like item ID, dining commons code, meal, date, stars, review text, status, moderator comments, and user ID.
  - API Endpoints:
    - Added endpoints for:
      - Retrieving all dining commons (`/api/diningcommons/all`)
        - Returns a json array of all current commons names and their codes.
- ```
[
  {
    "name": "Carrillo",
    "code": "carrillo"
  },
  {
    "name": "De La Guerra",
    "code": "de-la-guerra"
  },
  {
    "name": "Ortega",
    "code": "ortega"
  },
  {
    "name": "Portola",
    "code": "portola"
  }
]
```
- Retrieving meals for a specific dining commons on a given date.  
(`/api/diningcommons/{date-time}/{dining-commons-code}`)
    - Given the date/time and dining commons code, returns a json array of all valid meals for that dining common on that date.

```
[
  {
    "name": "Breakfast",
    "code": "breakfast"
  },
  {
    "name": "Lunch",
    "code": "lunch"
  },
  {
    "name": "Dinner",
    "code": "dinner"
  }
]
```

- Retrieving menu items for a specific meal at a dining commons on a given date.

(/api/diningcommons/{date-time}/{dining-commons-code}/{meal})

- Given the datetime, dining commons code, and meal, returns a json array of all valid menu items for that meal in that dining common on that date. Menu items are also all stored in the menu items database table if they do not exist in the table already.

```
[
  {
    "id": 31,
    "menuItem": {
      "id": 31,
      "diningCommonsCode": "portola",
      "meal": "lunch",
      "name": "Spicy Kale Caesar (v)",
      "station": "Greens & Grains"
    }
  },
  {
    "id": 32,
    "menuItem": {
      "id": 32,
      "diningCommonsCode": "portola",
      "meal": "lunch",
      "name": "Chicken Chipotle Wrap",
      "station": "Greens & Grains"
    }
  },
  {
    "id": 33,
    "menuItem": {
      "id": 33,
      "diningCommonsCode": "portola",
      "meal": "lunch",
      "name": "Spicy Kale Caesar (v)",
      "station": "Greens & Grains"
    }
  }
]
```

- Posting new reviews (/api/reviews/post).
  - Include the item\_id, a rating (between 1-5), comments, and a date served
  - Item\_id is cross referenced with menuItems table to check if the item\_id exists or not.
  - Admins can access other fields that aren't available to users (status, userId, moderator\_comments, moderator\_id, created\_date, and last\_edited\_date)

```
{
  "id": 0,
  "item_id": 0,
  "date_served": "2024-12-05T00:38:41.638Z",
  "status": "string",
  "comments": "string",
  "rating": 0,
  "userId": 0,
  "moderator_comments": "string",
  "moderator_id": 0,
  "created_date": "2024-12-05T00:38:41.638Z",
  "last_edited_date": "2024-12-05T00:38:41.638Z"
}
```

- - Retrieving all reviews for a logged-in user (`/api/reviews`).
    - Returns all reviews associated with the user, based on `userId`
  - Retrieving all reviews (`/api/reviews/all`)
    - Returns all reviews, regardless of `userId`.
- Integration with UCSB Dining API:
  - Services created to fetch dining commons names, meals, and menu items.
  - MenuItem data is saved in the local database to improve performance and allow for item reviews.
- Authentication and Authorization:
  - Implemented user authentication to secure review submission and retrieval.
  - Admin routes added for moderation purposes.

### Frontend Enhancements:

- React Components:
  - Created components for:
    - DiningCommonsTable
    - MealsTable
    - MenuItemPage
    - ReviewsTable
- Routing Improvements:
  - Dynamic routing implemented to handle pages like `/diningcommons/:code`, `/diningcommons/:code/:date`, and `/diningcommons/:code/:date/:meal`.
- State Management:
  - Utilized React Hooks and Context API for state management across components.
- Testing and Storybook:
  - Added comprehensive unit tests for all new components and services.
  - Storybook stories created for UI components to aid in development and documentation.

Developer Notes:

- Database Migrations:
  - Run new migrations to update the database schema with `Review` and `MenuItem` tables.
- Dependencies:
  - New dependencies added:
    - `org.json:json` version `20240303` for JSON handling in the backend.
    - Ensure you run `npm install` and update your Maven dependencies.
- Repository Methods:
  - Updated repository methods to handle `userId` fields correctly (e.g., `getByUserId`).
- Environment Variables:
  - Update any necessary environment variables for API keys and database connections.
- Error Handling:
  - Improved error handling for API calls and database operations.
- Code Cleanup:
  - Removed deprecated files and components related to UCSB Dates and Restaurants.