Mi-12 Assignment Requirement

assignment12_category_0013

🚩: 0 [If we have any update we will mention it here]. Check frequently to see if any updates have been made.

Objective

You must develop a **Hostel Management site for a University so that, they can serve Meals for their students** using the MERN stack.

Main Tasks

*****Make the homepage responsive for mobile, tablet, and desktop*****

*****Implement tanstack query in all the data fetching functionality (For GET method only) *****

HomePage:

1. **Navbar** has a **logo+website name, Home, Meals,Upcoming Meals Notification icon,** and **Join US (when not logged in)** button. If the user is logged in, his/her profile picture should appear on the navbar.

   If the user clicks on the **profile picture**, a drop-down will appear with the following items: **User name (not clickable), Dashboard,** and **Logout** button.

2. **Banner section** - A slider/banner/ a meaningful section. Inside the banner, there will be a **Heading Title, a Short Description,** and a **Search** Input Field with a button.

3. **Meals by category** - Implement a tab system for the **Meals by Category** section. There will be only three categories such as:

   a. **Breakfast**
   b. **Lunch**

   c. **Dinner**

   Explore React-tabs, or you can implement your own tab system. There will be only 4 tabs. Each tab will show a minimum of 3 meal cards. There will be a tab named **All Meals**, if the user clicks here, he will see all category meals. Implement the Active Tab feature.

   Each meal card will show **title,image,rating,price and details button.** On clicking the **details button (`.../meal/${_id}`)**, the user will be redirected to the meal details page.

   There will **see all** button. Clicking the button will redirect the user to the **Meals** page

4. **Add One/two extra sections relevant to this site.**

5. **Membership section:**
   Here user will see **three cards** to upgrade premium to request meals. Such as:
   a. **Silver**
   b. **Gold**
   c. **Platinum**
   Set different prices for each package.

NOTE: on clicking each package card, the user is redirected to the **(`.../checkout/${package_name}`)** page for the payment process.

6. **Footer section:**
   Make a relevant footer for your website.

Meal Detail Page:

7. Clicking on a meal/search result, the user will be redirected to the page (**`.../meal/${_id}`**) where it will show the meal details. The meal details will have the following:
    - Meal image
    - Admin/meal_distributor name
    - Meal Description
    - Ingredients
    - Post Time (store post time while adding a meal)
    - Rating
    - Like button
    - Meal request button
    - reviews

    Like button(require login):
    For logged-in users, Clicking the Like button will increase the reaction count to +1 and make the button look like the user loves it.

8. **Meal request button(require login):**
    a. Clicking this button, the logged-in user can request the meal(otherwise ask the user for login, using a modal/alert).
    b. You have to make a post request to handle this operation and save the meal and user information to the database.
    c. By default, while requesting the food, its status will be **pending.**

    ****If a user does not hold any packages, he/she can't request any meals.****

    Users can see his/her requested food on the **dashboard/my requested meals** page.

9. Below the meal details, there will be a review section. A user needs to log in before making a review and give a Like reaction. All users can see reviews for this meal. After giving a successful review, build a system to increase the review count for the meal.

    *Users can see his/her own reviews on the **dashboard/my reviews** page.

Note: Any user **can** make more than one review.

Meals Page:

10.

a. Show all meals on this page.
b. You have to implement a **search** functionality based on **meal_title.**
c. Implement **Filter-by-category** and **Filter-by-price-range** options.
d. implement infinite scrolling in the page, this means as you scroll the page more meal-cards will appear at the bottom

Upcoming Meals:

11.

a. Show all upcoming meal as cards added by the admin
b. User can give Likes to each meal. (a user can give 1 like on each meal)

Checkout Page(Private Route):

12.

a. This page will be a dynamic route and also a private route. Users will purchase the specific package.
b. Show some details about the package and implement **the Stripe Payment** method to purchase the package. After successful payment, show a modal that confirms the purchase.

*After successfully purchasing a package, the user will receive a **Badge** based on the package he/she purchased. **For example:** a user will receive a **Gold Badge** after purchasing a **Gold Package.**

Join US Page(Login/Register):

13. This is the page to implement authentication. Users will see a login form and add a link that will redirect the user to the Register page. Add at least one social login in both the Join Us and Register pages.
*When saving a user for the very first time, by default he/she will receive a **Bronze Badge**

Implement react-hook-form in the registration & login page.

Note: Do not enforce the email verification method and forget & reset password method, as it will inconvenience the examiner. If you want, you can add it after receiving the assignment result.

User Dashboard (Private Route):

Note: This must be a dashboard layout

14. When a user clicks on the Dashboard, he/she will be redirected to a page where there will be the following routes:
    A. My Profile
    B. Requested Meals
    C. My Reviews

15. **My Profile:**
    This page will have the user's name, image, email, and badges.

    There will be **two** badges and these badges will be visible only on the My Profile page when the conditions are fulfilled:

    1. **Bronze Badge:** If a user registers on the site, he/ she will receive the Bronze badge.
       Or,
    2. **Gold Badge:** If a user becomes a member by purchasing a premium package (**see req 5**), he/ she will be rewarded the specific badge.

    Note: You can use a picture/icon for the badge. It is up to you. Keep it relevant.

16. **Requested Meals:**
    If a user visits this page, he/she can see all the meals he/she requested/ordered. Show them in tabular form. Each row will have:

    - Meal Title

- Number of Likes/Likes count
- Number of Reviews count
- Status (**pending/delivered**)
- Cancel Button

Note: **sort** data based on **status** before show on the UI

17. **My reviews:**

    If a user visits this page, he/she will be able to see all the reviews he/she gives. Show them in tabular form. Each row will have:

    - Meal Title
    - Number of Likes/Likes count
    - Number of Reviews count
    - Edit Button
    - Delete button
    - View Meal Button (clicking the button will redirect to the specific meal detail page)

    *Handle the review update system by clicking the update button and make the delete button functional.

Admin Dashboard (Private Route):

Note: This must be a dashboard layout

18. When an admin clicks on the Dashboard, he/she will be redirected to a page where there will be the following routes:
    - A. **Admin Profile**
    - B. **Manage Users**
    - C. **Add meal**
    - D. **All meals**
    - E. **All reviews**
    - F. **serve meals**
    - G. **Upcoming meals**

19. **Manage Users:**

    Show all the users in a tabular form where each row will have:

    - User name

- User email
- Make admin
- Subscription Status(Membership)

*The admin can make a user admin by clicking on the **Make Admin** button. Implement a server-side search functionality to find a specific user (via **username and email**).

20. Add Meal:

This page will have a form with the following fields:

- Meal title
- Meal type/category (breakfast/Lunch/Dinner)
- Meal image
- Ingredients
- Description
- Price
- Rating
- Time/Date (post time)
- Likes(reaction, by default 0)
- Reviews (by default 0)
- Admin/distributor Name
- Admin/distributor Email
- **Add meal** Button
- **Add to Upcoming** Button

Implement react-hook-form on this page.

*Clicking the add meal button will post the data to the database. Show a toast/sweet-alert after successfully adding data.

*Clicking the **add to upcoming** button will post the data to the database and save to **upcoming collection.** Show a toast/sweet-alert after successfully adding data.

21. All Meals:

Show all meals in tabular form on this page. Each row will have:

- Meal Title
- Number of Likes/Likes count
- Number of Reviews count
- Distributor Name (who added this meal)

- Distributor Email(who added this meal)
- Functional Update button
- Functional Delete button
- View Meal Button (clicking the button will redirect to the specific meal detail page)

22.All reviews:

Show all reviews in tabular form on this page. Each row will have:

- Meal Title
- Number of Likes count
- Number of Reviews count
- Functional Delete button
- View Meal Button (clicking the button will redirect to the specific meal detail page)

Note: Implement a sort method based on likes count and reviews count

23. serve meals

On this page, the admin will see all requested meals by users. Show them in tabular form and each row will have:

- Meal title
- Email (who makes request)
- Name (who makes request)
- Status (pending/delivered)
- Serve Button

*By clicking the serve button will change the status of the meal from **pending** to **delivered.** If the meal is already delivered, show a toast as Already served.

*Implement a server-side search functionality to find a specific user (via username,email).

24. Upcoming meals

a. Show all upcoming meals in tabular form.
b. Sort them based on Likes count
c. There will be a **publish/production** button, when a meal holds up to 10 likes, clicking the button will add the meal to all-meals collection

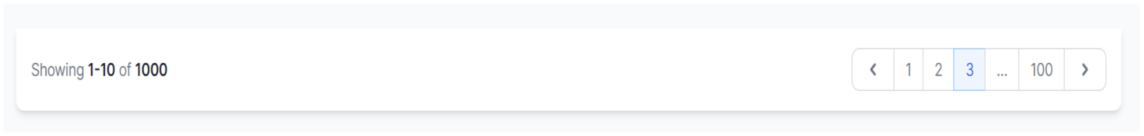
*****Make necessary commits and prepare a good readme.md file*****

Bonus Tasks

25. **Admin Profile:**
    This page will have the admin's name, image, email, and number of meals he added.

26. Implement JWT on login (Email/Password and social) and store the token.

27. Implement pagination at the footer of all the tables you have implemented (show 10 users/meals at a time). For example,



28. Hide **.ENV** credentials from both client and server git repositories

Optional Tasks

You have to implement **two tasks** from the following:

1) Implement the About Me section on the User's My Profile Page. This section will have an **Edit** button and clicking on the **Edit** button will render a form with the About Me field. On successful submission, the About Me information will appear on the About Me section of the My Profile Page.

2) Implement the react-awesome-button and React-select package.
3) Implement the react-modal package.

4) Implement Axios interceptor.

What to Submit

1. **Assignment Category:** assignment12_category_0013
2. **Admin email:**
3. **Admin password:**
4. **Front-end Live Site Link:**
5. **Client Side Github Repository Link:**
6. **Server Side Github Repository Link:**