Assignment_ID: assignment_category_0005

Video Guide: Link

Restaurant Management Website

A Restaurant Manager is seeking a web developer who can create a simple full-stack Restaurant website for the owner. You are required to create a website where a user can do the following tasks.

- See All Food Items
- See Single item
- Add A Food item
- Delete An Item
- Modify An Item

Make sure your website design is unique. Visit **ThemeForest**, **Dribble**, **Behance**, etc. to get some ideas. You can explore component libraries other than DaisyUI. Remember, a unique project will add more value to your portfolio.

🚩: 0 [If we have any update we will mention it here]

Main Requirements

1. Focus on making the website visually appealing. Ensure that
    - Color contrast is pleasing to the eye
    - The website does not have a gobindo design
    - The website has proper alignment and space
    - If needed, customize the design of any component you are taking from any component library. (For example, you are using Daisy UI & have taken a card component from Daisy, if needed, customize the styling of the card to make it reasonable rather than just copy & paste it.)

    Note: Your website can not be related to your previous assignments' layout/ design or any practice project shown in the course modules or our conceptual sessions. Ex: You can't copy any design or similar functionality/ layout of

    - **Career hub**

    - **Dragon news**
    - **Coffee store**
    - **Ema john**
    - **Car Doctor**
    - Any of your previous assignments or conceptual session projects. If any similarities are found, you will get **zero(0)** as a penalty.
2. Make sure to keep the **navbar** and **footer** on all the pages except on the **404 page**. Create a reasonable and meaningful footer. (including website **logo**, **name**, **copyright**, some contact information, social media links, address, etc.)

Navbar🧭:

Your website should have a navbar with the following information:

1. Website name/logo
2. Home
3. All Food Items,
4. Blog and,
5. Conditional login/logout
6. User Profile.

Note: The User profile picture on the navbar is conditional based on login. If the user is logged in, the navbar will show the profile picture and logout button; otherwise, it will show the Login button.

Homepage 🏠:

👉 **Banner section:** A slider/banner/ a meaningful section. Inside the banner, there will be a Heading Title, a Short Description, and a button that will redirect the user to the all menus page.

👉 **Top Food section:** Show 6 top-selling Food Items including the following pieces of information:

- Food Name
- Food Image
- Food Category
- Price
- Details Button

💡**hints:** Count the number of orders for a food everytime a user books a food. Depending on this count determine the top-selling project. By default, the count will be zero(0). I.e. while creating a new product store the count property.

👉 **see all button**: Below the 6 cards, there will be a see all button that will redirect the user to the all Food items page.

👉 **Extra section:** Add **2(Two) relevant** and **attractive** sections except the **nav**, **banner**, **footer**, and **Top Food section**.

All Food Page 🥑:

👉 **Food Cards:** Here you need to show all the food items stored in your database.

👉 **Search Functionality**: on the top of the food cards section you have to implement a search functionality based on the food name. You can implement a search option using either frontend or backend.

👉 **pagination:** At the bottom of the cards section you need to create a pagination. First, you have to load 9 cards of data from the database and then you will show the others based on the page number. Implement backend pagination.

👉 **Each Card**: Each card Item will have the following information:

- Food Name
- Food Image
- Food Category
- Price
- Quantity (see bonus part)
- Details Button

❇️ On clicking the details button will navigate to the Single Food Page.

Single Food Page 🍕:

Show details information about a single food item which will include the following:

- Food Name
- Food Image
- Food Category

- Price
- Made By (Who added the food)
- Food Origin (Country)
- A short description of the food item (for example: ingredients, making procedure, etc.)
- Order button

📝Note: clicking the order button will redirect users to the food ordering page. This page will be a **private route**. if the user is not logged in then it will redirect him/her to the login page and after successful login the user will redirect to the food ordering page.

Food purchase Page💸:

This page will have a form containing the following information:

- Food Name
- Price
- Quantity (more details see bonus part)
- Buyer Name (**Read-only**. This will be picked from the logged-in user's information)
- Buyer Email (**Read-only**. This will be picked from the logged-in user's information)
- Buying Date (see bonus part for details)
- A button to purchase the food item.

📝Note: on clicking the purchase button the information will be stored in the database. Also, On a successful order, you have to show a toast/alert **(do not use browser alert)**.

Blog Page🅱️:

Create a blog page where you have to answer the following questions:

1. What is One way data binding?
2. What is NPM in node.js?
3. Different between Mongodb database vs mySQL database.

Login and registration systems

Registration Page ➕:

Create a Registration page will have the Email/Password form having the following fields:

- Name
- Email
- Photo URL
- Password

📝Note: Make sure you add the user information in a MongoDB collection after successful registration and use them as per your need.

⚠️ *Do not enforce the email verification method, as it will inconvenience the examiner. If you want, you can add email verification after receiving the assignment result.*

Login Page ➕:

When a user clicks on the login button, they will be redirected to the login page which has the following:

- Email/Password
- A Social Login System
- A link that will help the user toggle the login and **registration page**

📝 Both Registration and Login pages, display relevant error (⚠️) messages when necessary.

My profile 🧑‍🏫:

On clicking on my profile *(it can be implemented using a dropdown when clicked on the user profile image.)* there will be three routes:

1. My added food items
2. Add a food item
3. My ordered food items

My added food items Page 🍒:

In this route, you have to show all food items added by the currently logged-in user (💡hints: You have to filter data based on the user's email). You can show all the food items in tabular/card. Each row/card will have:

➔ Some Food info (example: food img, name, price, etc)
➔ an update button/icon.

📝 clicking the update button/icon will redirect to the update page or open a **modal**. There will be a form that has the product info and an update button. When click the update button the product info will be updated. Don't let other users delete your added food items.

Add A food item Page 🍉:

This page will have a form having the following fields:

1. Food Name
2. Food Image
3. Food Category
4. quantity
5. Price
6. Add By (name & email: this info added from currently logged-in user.)
7. Food Origin (Country)
8. A short description of the food item (ingredients, making procedure, etc.)
9. Add Item Button

📝 on clicking the **add button** the information will be stored in the database. on successfully adding a food item, you have to show a toast/alert *(do not use browser alert)*

My order Page 🍉:

In this route, show all food items ordered by the logged-in user. You have to filter the data based on the logged-in user email. You can show all the food items in table/card. Each row/card will have:

1. Some Food info (example: food img, name, price, added time, food owner, etc)

2. a delete button/icon which will help the user to delete the ordered item from the list.

404 Page ❌:

Create a 404 page. Add any interesting jpg/gif on the 404 page. Do not add header & footer on this page. Just add a jpg/gif & a Back to Home button. The Back to Home button will redirect the user to the home page.

Bonus Requirements

1. Single Food Quantity💰

📌 You have to implement a feature on the food purchase page where the feature will be: if the available quantity of a food item is zero then you have to show a message to the buyer that he/she can not buy that item because that item is not available.
📌Also the buyers can't buy more than available quantity. (assume that a food item has 20 quantities then a user won't be able to buy more than 20 quantities).
📌Don't let the user purchase his/her own added food items.

2. Commits & readme ⚙️

📌 Minimum 15 meaningful git commits on the client-side repository.
📌 Minimum 10 meaningful commits on the server-side repository.
📌 Create a readme on your client side and add the following info:
❇️ Your project name & live link.
❇️ 5 main Features of your projects with bullet points

3. Fix your Reload & Error Issue 🔄

📌 If you reload the protected/private routes (after login), it will not redirect the user to the login page.
📌 If you reload the website, it will not show the site not found. (Netlify & surge).

4. Make HomePage responsive 💻

📌 Make the Homepage fully responsive (mobile, tablet, and desktop)
📌 Optional: if possible make the whole website responsive.

5. Website Naming

📌 Give your website a name. The website title will be changed according to your route/page. Suppose your website name is PHero. Then, on the 'login' router/page, your website title will be 'PHero | Login'.
💡**hints**: explore the react-helmet npm package

6. Authorize your server routes with the JWT token. 🔐

Upon login, you will create a JWT token and store it on the client side. You will send the token with the call and verify the user. Implementing 401 and 403 is optional. Ensure you have implemented the JWT token, create a token and store it on the client side for both email/password-based authentication and social login. You must implement JWT on your private routes.

7. Environment Variables 🪪

📌 Use the Environment variable to hide the Firebase config keys and Mongodb credentials.

Optional (But Highly Recommended)

Implement any two tasks from the following optional list:
👉 Add a spinner when the data is in a loading state. You can add a gif/jpg, use any package, or customize it using CSS.
👉 Explore and implement any of the animations from the Framer Motion.
👉 Explore and implement Tanstack query mutations in your api.
👉 Add multiple filtering system on all food items page. The filtering system should be implemented on the server side. Think about mongodb $and , $or operators.

What to submit

1. generated client-side GitHub repository
2. generated server-side GitHub repository
3. Generated live site link.

Some Guidelines:

1. Do not waste much time on the website idea. Just spend 15-20 minutes deciding, find a sample website, and start working on it.
2. Do not waste much time finding the right image. You can always start with a simple idea. Make the website and then add different images.
3. Divide the whole project into several parts and work one task at a time.
4. Stay calm, think before coding, and work sequentially. You will make it.
5. Be strategic about the electricity issue.
6. use chatGPT to generate JSON data.