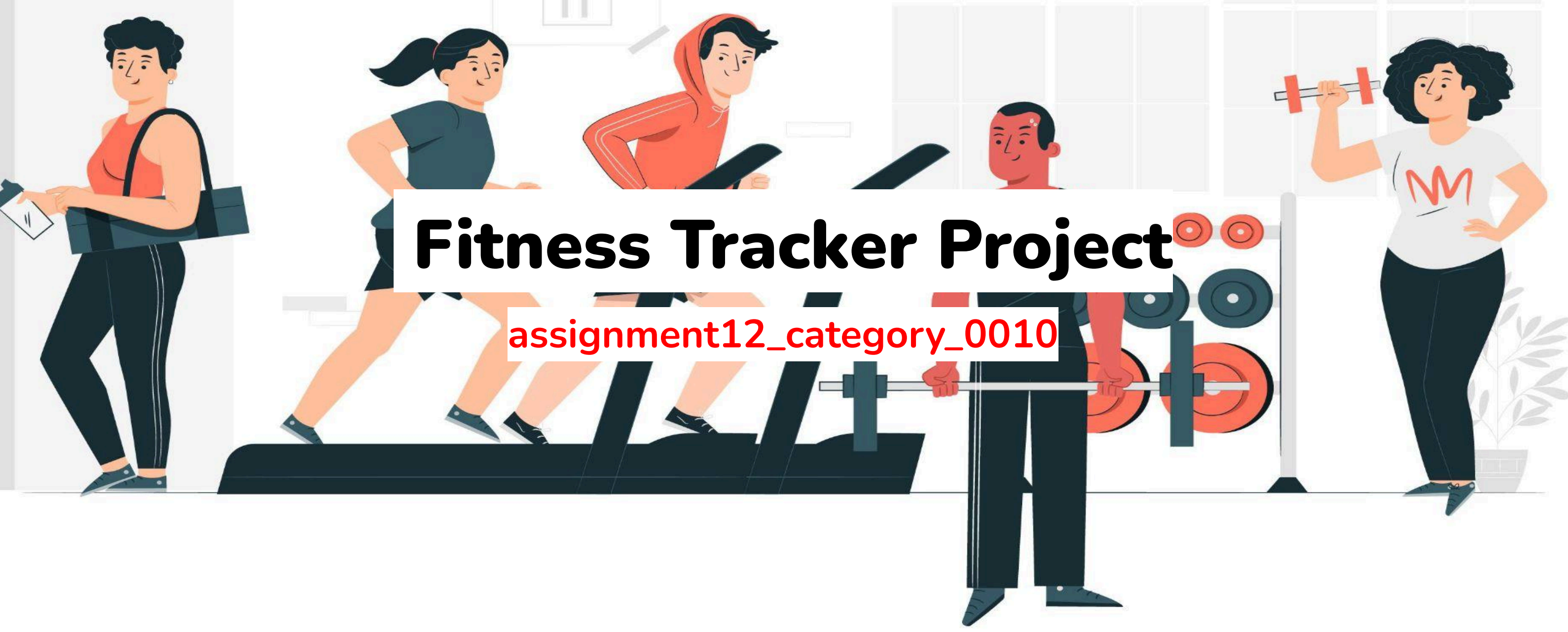

Fitness Tracker Project

assignment12_category_0010

About the Company:

Our company is at the forefront of revolutionizing the fitness industry. We are dedicated to providing innovative solutions that empower individuals to lead healthier, more active lifestyles. Combining cutting-edge technology with a passion for fitness, we strive to create a community-driven platform that inspires, motivates, and supports our users on their wellness journey.

Job Overview:

They are seeking a skilled MERN Stack Developer to lead the development of a cutting-edge Fitness Tracker platform. Using MongoDB, Express.js, React.js, and Node.js, you'll create an immersive experience for users to track fitness progress, set goals, and engage with a vibrant community. Your role will be pivotal in shaping fitness technology's future, optimizing user experience and internal processes. Join us in revolutionizing the way people approach fitness and wellness.

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 [changes will be tracked here]

🚀Key Rules🚀

1. Commits & readme ⚙️

📌 Minimum 20 meaningful git commits on the client side.
📌 Minimum 12 meaningful commits on the server side.
📌 Add a meaningful readme.md file with the name of your website, Admin username, password, and live site URL. Include a minimum of 10 bullet points to feature your website.

2. **Make HomePage responsive** 💻
   📌 Make the website fully responsive (mobile, tablet, and desktop)

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. **Environment Variables** 🔐
   📌 Use the Environment variable to hide the Firebase config keys and Mongodb credentials.
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. **Lorem ipsum** :
   📌 Don't use any Lorem ipsum text; you can not use the default alert to show any error or success message.
7. **Toast/Sweet Alert** :
   📌 Show sweet alert/toast/notification for all your **CRUD** operations, successful authentication login, and sign-up. Don't use the default browser alert.
8. **Data fetch(Get)** :
   📌 Implement **tanstack query** in all the data fetching functionality (For **GET** method only)
9. **Total Role of this project (3)**:
   📌 Admin
   📌 Trainer
   📌 Member (default)

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**
- **Bistro boss**
- 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 page.
3. All Trainer Page.
4. All Classes page.
5. ***Conditional*** Dashboard.
6. Community/Forums page.
7. ***Conditional*** login/register.
8. User Profile.

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

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**

📝***Note***: Both Registration and Login pages, display relevant **Error** (🛑) messages when necessary. Also, A user by default will get the **(role: member)** when he/she registers or sign in with a popup.

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 **classes** page.

👉 **Featured section:** Create a featured section that highlights key website features through a series of cards. Each card should include a concise title, an engaging description, and an icon or image that visually represents the feature.
💡For ideas visit this but don't copy this design 🔗**link**

👉 **About section:** this section will have some info about the organization. You can use any type of layout or design but try to match it with your whole website design.

👉 **Featured classes:** Create a "Featured Classes" section showcasing the top six most booked classes, based on their total booking counts. Each class should be displayed with a title, a brief description, and the **total number of bookings** to highlight its popularity. All data will be from the database.
📝 Use the backend and MongoDB $sort method to filter data based on the total booking number. For that, you must count the total booked each time a user booked a class.

👉 **Testimonials or Reviews:** Implement a carousel slider that displays three review cards simultaneously. This slider should be capable of cycling through all the reviews submitted by a **member**. The slider will include functionality to navigate**(next, prev)** through the reviews smoothly, allowing users to view each one in succession. See this demo

👉 **Latest Community/Forum posts:** show **recent** six(6) community forum posts or articles. Ensure to include direct links for further reading or exploring the blog section. Aim to feature 4-6 posts in total.

👉 **Newsletter section:** Design a subscription section that includes fields for the user's name and email address, along with a "Subscribe Now" button. When the "Subscribe Now" button is clicked, the user's information should be saved to the database. Ensure that this process does not require users to log in to subscribe to the newsletter.

👉 **Team section:** Display at least three trainer profiles, each accompanied by key information. Each profile should include the trainer's name, a brief biography, areas of expertise, and a professional photo.

All Trainers Page👯 (Public)

👉 **Trainer Section:** Show all the Trainer profiles. Each Card contains the following info.

- Trainer name
- Profile Image
- Years of Experience
- Social icons
- Available slots.
- Other info if needed.
- **Know more**

📝By clicking the ***Know More*** button should redirect to the ***Trainer details page***,

Trainer Details Page🤵 (Public)

Design a Trainer details page with a layout consisting of two sections. 👇

Trainer info	**Available slots**

1. **Trainer Information Section:** Include the trainer's name, photo, details, expertise, and other info. This section should provide comprehensive information about the trainer's background and qualifications.
2. **Available Slots:** Display a schedule of the trainer's available time slots for booking sessions. Each slot will be represented as a button/link.

🚀clicking any **available slot** will redirect the user to the **trainer booking** page.

👉**Be A Trainer Section:** Below/above the ***Trainer details*** *and* ***Available Slots sections***, there will be a"**Be A Trainer**" section. This section should feature a prominent Call to Action (**CTA**) button labeled "**Become a Trainer**." Upon clicking this button, users will be redirected to the "**Become a Trainer**" page.
(*search this keyword on Google for better understanding*: keyword)

Trainer Booked Page👨‍💻(Private)

👉 From the Trainer Details Page **Available Slots** when a user clicks on any available **slot**, they will be redirected to the **Trainer Booked Page**.

This page will show the following info:

- Trainer name
- Selected slot
- Classes
- packages: (ex: **Basic**, **Standard**, **Premium**)
- **Join now button**

As for packages will include some extra benefits for example:

Basic Membership	Standard Membership	Premium Membership
Access to gym facilities during regular operating hours.	All benefits of the basic membership.	All benefits of the standard membership.
Use of cardio and strength training equipment.	Access to group fitness classes such as yoga, spinning, and Zumba.	Access to personal training sessions with certified trainers.
Access to locker rooms and showers.	Use of additional amenities like a sauna or steam room.	Discounts on additional services such as massage therapy or nutrition counseling.
Price: $10	**Price: $50**	**Price: $100**

Users are **required** to choose one membership plan from the options provided. The table design and content are subject to customization per the assignment requirements.

👉 Click the **Join Now** button to redirect the user to the **payment page**👇

<u>Payment Page</u>💸(Private)

👉 On this page, you have some info:

- ❖ Trainer name
- ❖ **Slot** name
- ❖ Package name
- ❖ Price
- ❖ Your name, email & other info
- ❖ Other info if needed
- ❖ **Confirm Button**

📝 Clicking on the **confirm button** will save all the information in the **database**.

⏰ here you can use a **form** or **card**. If you add a **form** don't let the user **modify** any of the above info.

<u>Be a Trainer Page</u> (Private)

👉 This page will have a form where the user can submit the following info:

- Full Name
- Email **(read-only)**
- Age
- Profile Image
- Skills (use checkbox for selecting multiple skills)
- Available days a week. (Ex: **Sun, Mon, Tues, Thu**) see below for more info👇
- Available time in a day
- <u>Other info depending on your project.</u>
- **Applied Button**.

📝 Also, the **status** will be **pending** by default. **All the info will be stored in the database by clicking the apply button**.

💡For **Available days in a week** you have to use this **<u>npm package</u>**: <u>React Select</u>. Also, the user can't insert any value. There will be only 7 days' name on the select option.

🛑 Users don't have the option to add his/her input for a week. user only can select the item (7 days' name) from the select box.

Classes Page(Public)

👉 **All Classes Section:** Create an **All Classes** section where users can access a comprehensive list of all available classes.
📝 Each class will display all necessary information, including class description and any additional details. Additionally, there will be a ***list of trainers who specialize in teaching that specific class, a*** maximum of **five(5**) trainers.

For example, if the class is "**Yoga**," there will be a list of trainers who conduct Yoga sessions. See this demo

👉 **Pagination:** Implement **pagination** for the "All Classes" section, displaying **six** classes per page by default. Users can navigate through the list of classes by changing the page number, allowing them to view the remaining classes.

💡From the list of trainers associated with a specific class, if a user clicks on any of the trainer's images, will redirected to that particular ***Trainer details page***.

Forum Page🅱️(Public)

📖**Post section:** Display 6 posts per page on the forum page, implementing **pagination** using **backend** functionality.

👉 **Voting System:** Implement an up-vote and down-vote system similar to Quora's. Users need to log in to **vote for** forum posts,

Dashboard 🪪(Private)

👉 In the dashboard, the visibility and access to pages will be conditional based on the user's role. Each page within the dashboard will be private and restricted based on the user's role. For instance, an **admin** won't be able to see the "**Manage Slots**" page, while a trainer won't have access to the "**All Trainers**" page. Similarly, members won't be able to view either of these pages. Also, the primary **Navbar** and **Footer** won't be shown in the **dashboard**.

Admin 🤵 (Private)

An admin will see The following routes/pages 👇

- ❖ **All Newsletter subscribers:** here you have to show All newsletter subscribers in tabular format.

- ❖ **All Trainers:** On this page, you have to show all Trainers in a tabular format. Where she can do the following:
  👉 **Delete a Trainer:** The Delete Trainer button/icon enables the **Admin** to remove a user from the ***Trainer role***. When this action is performed, the user's role will be changed back to **Member**, and they will no longer have access to Trainer-specific routes/pages.

- ❖ **Applied Trainer:** On this page, the admin will see all the applications for Trainer. You can show tabular/card.

  👉 There will be a details button that will redirect to a detailed page showing specific information about the applied trainer. This page will display comprehensive details about the applicant. There will be two additional buttons on this page: one for confirming the application and another for rejecting it.

  👉 **confirmation button:** On clicking the **confirmation** button the applied person's status will be changed from normal user to Trainer and that particular entry will be removed from the **Applied Trainer.**

  🛑**reject button:** *See the challenging section* 👇

- ❖ **Balance**: on this page,
  👉 Admin will see the overview of their financial activities. This includes displaying the **Total Balance** and the last **six transactions** related to **booking payments**.
  📝 The Total Remaining Balance is calculated by summing all booking payments made by the member.
  👉 Additionally, the last six booking payment transactions are listed, providing a recent history of financial activities.
  👉 Add a **pie chart/bar chart** or any other chart where you have to show total **newsletter subscribers** vs **total paid members**.

- ❖ **Add new Class:** On this page, there will be a form where an admin can add a new class. Added class will be shown on the **Classes Page.** The form will include the following info:
  - ➢ Class name
  - ➢ Image
  - ➢ Details
  - ➢ Other additional info if needed
  - ➢ submit/Add button

📝 clicking the submit/add class button will store the info in the database.

Trainer 👯 (Private)

A Trainer will see The following routes/pages 👇

- ❖ **Manage Slots:** on this route, the trainer will see all slots in tabular format.
  👉 If the slot is booked trainer will see that particular slot info including who booked that slot.
  👉**Delete a slot**: Each slot will have a delete button/icon. When the delete button/icon is clicked, a confirmation prompt will appear to ensure the user wants to delete the slot. Only after confirming will the slot be removed from the database.
- ❖ **Add New slot:** On this page, there will be a form that includes the following fields:
  - ➢ All his/her previous data when he/she applied to be a trainer will be **read only**
  - ➢ Select days: he/she previously added will be shown. use this **npm package**: React Select.
  - ➢ Slot name (example: morning slot)
  - ➢ Slot time (example: 1 hour)
  - ➢ Classes include (This will display all classes added by the **admin**. Trainers can select one or more classes from this list, but they will not have the ability to add new classes themselves. You can use the checkbox or his **npm package**: React Select.)
  - ➢ Other info if needed.
  - ➢ submit/add class button

📝 clicking on the submit/add class button will store the info in the database.

❖ **Add new Forum:** both **admin** & **trainer** will get access to this page. Both of them can add a new forum. All added forums by the admin and trainer will be shown on the **Community/Forums** page. **[more see bonus part]**

Member 👩‍🎓(Private)

A member will see The following routes/pages 👇

❖ **Activity Log page:** On this page, the status of each member who has applied for a trainer role will be displayed. Initially, their status will be marked as "**Pending**." If an applicant is rejected, their status will change to "**Rejected**." Alongside this status, an **eye icon** will appear.

👉**Eye Icon Action:** Clicking the e**ye icon** will open a modal. This modal will display the rejection message or feedback provided by the **admin**.

📝***Note***: Successfully applied trainers' information will not be displayed on this page. Once an applicant is approved by the **admin,** their role will be changed from ***member*** to ***trainer***, they will no longer have access to this route/page.

❖ **Profile Page:** The Profile page allows users to manage their account details comprehensively. Users can modify various aspects of their account information, except for their email address, which remains uneditable. Info includes with:

★ Name
★ Profile Picture
★ Email
★ Last login status (get it from firebase user object)
★ *Other info if needed*

❖ **Recommended Classes Page:** on this page, we'll add a component that displays a list of recommended classes for users. The recommendation logic can be based on various criteria such as popularity, user preferences, or admin-selected classes. ***You have the right to implement it on your choice***.

Challenging Task 🎁

🚀Search Functionality (All Classes Page):

Implement a search feature allowing users to find classes by name on the "**All Classes**" page. The search should be case-insensitive, ensuring results regardless of case variations. Watch this video: link

🚀Community/Forums page:

For each post, you have to show the ***admin/trainer*** Badge for each of their posts. What you implement it's totally up to you. You can get inspiration from Facebook group **admin** and **moderator** posts. Or you can use an **icon** to show the Badge**.**

🚀Admin Reject Trainer Action:

Upon clicking the **Reject** button/icon, a modal will display containing the applied trainer's information. This modal provides details of the applicant's profile, with the feedback textarea field being editable for providing rejection feedback. After submitting the modal, the entry corresponding to the applied trainer will be removed from the "Applied Trainer" section.

🚀Private Routes Authorization 🔐

Upon login, you will create a **JWT token** and store it on the client side **localStorage**. You will send the token with the call and verify the user. Implementing **401** and **403** for invalid/unauthorized access. 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 all of your **private routes** to secure your API access.

⏰Don't use cookies to store JWT tokens. Use Local Storage to store JWT Tokens.

What to Submit

- ☐ Client-side Repo Link
- ☐ Server Side Repo Link
- ☐ Generate client live link
- ☐ assignments variants/category
- ☐ ***Admin credentials (email, password)***

Resources🔗

➔ Data-flow Dia-gram: link

➔ React select
➔ Tanstack query
➔ Tailwind css component libraries: react-flowbite, mamba UI, headless UI, Material Tailwind, wind UI

📝 Note: Before integrating any of the mentioned Tailwind CSS component libraries into your project, ensure their compatibility and stability. Most libraries are stable, but verifying their performance within your specific project is crucial. Configuration might pose challenges for some libraries, so thorough research and testing are recommended before starting. You can use other libraries except **Daisy UI**.

Guidelines

★ It's a very very big project so don't try to solve all of your problems at a time. Break a big functionality into many little pieces then try to solve them one by one.
★ Do not waste much time on the website idea. Just spend 15-20 minutes deciding, find a sample website, and start working on it.
★ Divide the whole task into 3 parts: easy, medium, and hard. Keep easy tasks between your medium and hard tasks. When you get stuck on difficult tasks, do the easy ones. Then again try to solve difficult, medium tasks.
★ Don't do coding for a very long time. Take a 5-10 minute break every 40/50 minutes. Give your brain a break to think of new ideas for your task.
★ Be strategic about the electricity issue.
★ Just don't copy code from ChatGPT, try to understand how it works and how you can implement it in a better way.

🥰All The Best 🥰