assignment_category_06

Congratulations on successfully clearing the first selection round! We were truly impressed by your application and skills. As you move forward to the next stage of our selection process, this is your opportunity to demonstrate your problem-solving abilities, showcase your creativity, and tackle challenges with innovative solutions. We look forward to seeing your unique talents shine as you deliver exceptional and high-quality work. Best of luck! 🚀🍀

Project Theme: Marathon Management System

The Marathon Management System is a platform that helps organize marathon events by connecting event organizers with participants. Users can create marathons, sign up for events, and manage their registrations through a personal dashboard. This project offers practical experience in building full-stack applications, setting up user login, managing data, and connecting with a secure database.

The website should allow users to manage and explore marathon events through various features:

User Capabilities:

✅ **Add/Update/Delete Marathon Events:** Users can manage their marathon events, including updating or removing them.
✅ **View Marathon Details:** Users can browse events, view detailed information, and see recommendations.
✅ **Apply, Delete, Update:** Logged-in users can apply for marathons, update their applications, or delete them if needed.

Key Rules:

- Include a minimum of 18 notable GitHub commits on the client side
- Include a minimum of 8 notable GitHub commits on the server side
- Add a meaningful readme.md file with the name of your website and a live site URL. Include a minimum of five bullet points to feature your website.
- Make it responsive for all devices. You need to make it responsive for mobile, tablet and desktop views.

- After reloading the page of a private route, the user should not be redirected to the login page.
- Use the Environment variable to hide the Firebase config keys and Mongodb credentials.
- Don't use any Lorem ipsum text; you can not use the default alert to show any error or success message.

Main Requirements

1. Layout

Navbar:

- **Before Login:** Left side: Logo and Right side: links Marathons, Login, and Register.
- **After Login:** Shows Logo, Marathons, Dashboard, User Avatar, and Logout Button.

The main section is dynamic, displaying content based on the selected route. It updates automatically as users navigate through different pages of the application.

Footer:

- Website Logo/Name, Description, Copyright Information, and Useful Links.

Example of Main Layout Design

Header
Dynamic sections based on Routes
Footer

Sample Dashboard Layout Design

Header	
Route/Links	Content/Pages

Add Marathon	Maratho form
My Marathon List	List of marathons (Fetched from Database)
My Apply List	List of apply (Fetched from Database)

Footer

2. Home Page (Public)

The home page will be visible to all users and includes the following sections:

Banner Section:

- A simple image slider/carousel with at least three slides containing meaningful text or event highlights.

Marathons Section:

- Show **6 marathon cards** using **MongoDB's limit() method**. Each card should display:
  - Marathon Title, Location, Registration Dates, and a **"See Details"** Button.

🎯 **When the user clicks the "See Details" button, they are redirected to the marathon details page.**

Upcoming Marathons Section:

- Displays six randomly upcoming marathon event cards with important dates and details. This section highlights upcoming events and provides users with a quick overview. (**Static content**)

Extra Section

- Add 2 relevant and meaningful extra sections that are mentioned above on the Home page.

3. Login Page: When you click the login button on the navbar it redirects to the login page. You have to use a password and email-based authentication to log in. The login page will have-

- Email
- Password
- Google login/ GitHub- implement any of one
- A link that will redirect to the Register page

🎯**Here the email and password should match with the registered email and password. If it doesn't match, show an error message. You can show an error by using toast/sweet alert if you want.**

4. **Register Page:** You have to use a password and email-based authentication to register. The Register page will have the following -

- Name
- Email
- photoURL
- password
- A Link that will redirect to the login page

★ For password verification you need to follow this -

- Must have an Uppercase letter in the password
- Must have a Lowercase letter in the password
- Length must be at least 6 character

★ If any of this isn't fulfilled it will show an error message /toast

★ After successful login or Register you need to show toast/sweet alert

🎯**Don't implement email verification or forget password method as it will inconvenience the examiner. If you want, you can add these after receiving the assignment result.**

5. Add Marathons Page (Private Route)

This page allows logged-in users to create marathon events.

Fields to Fill:

❖ **Marathon Details:**

➢ Marathon Title

➢ Start Registration Date

- ➢ End Registration Date
- ➢ Marathon Start Date
- ➢ Location
- ➢ Running distance
- ➢ Description
- ➢ Marathon Image
- ➢ createdAt = new Date()
- ➢ The **Total Registration Count** should have an initial value of **0**, indicating no registrations when a marathon is first created.

❖ **Running Distance:** Choose options like 25k, 10k, or 3k using a dropdown menu.

❖ **Date Picker:** Use the react-datepicker library for start and end registration dates and marathon start date.

What Happens Next:

- After clicking the **Submit** button, show a success message and store the marathon details with user information to the database.

6. Marathons Page (Private Route)

This page displays all created marathons in a **3-column grid layout** using cards. Each card will display essential marathon information, such as:

❖ Marathon Image
❖ Marathon Title
❖ Marathon Location
❖ Registration Start and End Dates
❖ **"See Details" Button**

When the user clicks the **"See Details"** button, they should be redirected to the Marathon Details page, where they can view more information about the selected marathon event.

7. Marathon Details Page (Private Route)

Displays detailed information about a specific marathon also show total registration count.

Features:

❖ **Register Button:**

- ➢ Enabled if registration is open (between the start and end dates).
- ➢ Clicking it redirects users to the registration page.

Registration Form Fields:

- ❖ Email (auto-filled for logged-in users, also add marathon title and start date Readonly), First Name, Last Name, Contact Number, and Additional Info.
- ❖ The **Marathon Title** and **Start Date** are displayed based on the selected marathon. (Read Only)
- ❖ Upon successful registration, update the **Total Registration Count** by incrementing its previous value by **1**.
- ❖ After submission, save the registration details in the database and display them on the **Dashboard -> My Apply** page.

8. My Marathons List (Private Route)

This page displays marathons created by the logged-in user in **table format**.

Features:

- ❖ **Update Button:** Opens a form in a modal, allowing users to update the marathon information.
- ❖ **Delete Button:** Opens a confirmation modal to delete a marathon entry.

9. My Apply List (Private Route)

A logged-in user should see only the marathons they have applied for, displayed in a **table format**. This means the data shown should be specific to the currently logged-in user.

Features:

- ❖ **Update Button:** Opens a form in a modal where users can update their registration details.
- ❖ **Delete Button:** Opens a confirmation modal to remove the registration.

🎯 **Marathon Title** and **Marathon Start Date** should be **read-only** fields, ensuring users cannot modify them.

Additional

- **Dynamic Title:** Make your website title Dynamic. For every Route change, The Website Title will be changed based on that route.
- **404 page:** Add a 404 page/Not Found Page
- **Spinner:** Show a loading spinner when the data is in a loading state.
- **Toast:** For all the CRUD operations, show relevant toast/ notification/ sweet alert with a meaningful message

Challenges Requirements:

- **My Apply List - Search by Title:**
  - **Search Feature:** Users can search for marathons they have applied for based on the marathon title. A text input field will be placed at the top of the "My Apply List" page.
  - **Backend Query:** The search will be handled **server-side**. The backend will process the search input and filter marathons by location, ensuring case-insensitive matching. This will be achieved by using the **MongoDB** **$regex** method for case-insensitive searching.

🎯 Although developers can implement the search feature on the client side, using server-side search is **recommended** for better scalability, performance, and security.

- **Countdown Timer:** Shows how many days, hours, minutes are left until the marathon starts using the react-countdown-circle-timer.
- **JWT Authentication:** 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.
- Add a sorting option in the marathon list based on the **createdAt** field using MongoDB's **sort()** method. This allows users to view marathons from the newest to oldest or vice versa.

Optional (But Highly Recommended):

Implement any two tasks from the following optional list:

1. Try to use any other tailwind CSS library like - mamba Ui, shadcn, chakra UI, flowbite.
2. Add one extra feature of your own. This will help you in the future to differentiate your project from others.

3. Implement Pagination in the Marathon Page. Show 6-9 services per page.
4. **Dark / Light Theme:** Implement a dark / Light theme toggling in your Navbar for the whole web application.

What to submit:

1. Your client-side code GitHub repository
2. Your server-side code GitHub repository
3. Your live website link

Additional information:

1. You can host images anywhere.
2. You can use vanilla CSS or any library.
3. Try to host your site on Firebase (Netlify hosting will need some extra configurations)
   Firebase Hosting Setup Complete Issue
4. Host your server-side application on Vercel. If needed, you can host somewhere else as well.
   How to deploy a Node/Express server using Vercel CLI
   Some Common Vercel Errors
5. Make Sure you deploy server-side and client-side on the first day. If you have any issues with hosting or GitHub push, please join the "Github and deploy" related support session.

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. Don't look at the overall task list. Just take one task at a time and do it. Once it's done, pick the next task. If you get stuck on a particular task, move on to the next Task.

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. You can use chatGPT for other purposes as well.