assignment_category_05

Dear Candidates,
Are you a front-end developer who thrives on building beautiful and functional web experiences? Do you enjoy clean code and keeping up with the latest tech trends? We're looking for you! We're building the next generation of our blog website and need a talented developer to join our team. You'll collaborate with designers and back-end developers to bring mockups to life using React, Firebase, and other cutting-edge tools. You'll be responsible for crafting a user-friendly and engaging experience for our audience. This includes building dynamic web pages, ensuring smooth navigation, and optimising website performance for all devices.

Blog Website

Welcome to the Blog Website Development project! We're seeking a talented Front-End Developer to help bring our next-generation blog website to life. This exciting project will allow you to design and implement beautiful, responsive web pages, collaborating with designers and back-end developers. The project involves creating a dynamic, user-friendly experience for visitors, using modern technologies like React and Firebase.

Ensure the Following things to get 100% mark

- Include at least 15 meaningful commits on the client side & 8 meaningful commits on the server side with descriptive messages.
- Include a README file with the project name, purpose, live URL, key features, and any npm packages you have used.
- Ensure the website is fully responsive on mobile, tablet, and desktop.
- Secure Firebase configuration keys using environment variables.
- Secure your MongoDB credentials using the environment variable.
- Create a design that encourages recruiters. Color contrast should please the eye & ensure that the website has proper alignment, space and The website does not express gobindo design

Deployment Guideline

If your Deployment is not okay you will get 0 and may miss the chance of our upcoming rewards.

- Ensure that your server is working perfectly on production and not throwing any CORS / 404 / 504 Errors.
- Ensure that your Live Link is working perfectly and that it is not showing errors on Landing in your system.
- ⚠️ ensure that the page doesn't throw any error on reloading from any routes.
- ⚠️ Add your domain for authorization to Firebase if you use Netlify / surge
- ⚠️ Logged in User must not redirect to Login on reloading any private route

---

Layout & Page Structure

1. Create a navbar which will have the following links:
   a. Home
   b. Add Blog (Private)
   c. All blogs
   d. Featured Blogs
   e. Wishlist (Private)

Note: If the user is not logged in, the Login and Register buttons should show, and if the user is logged in show his profile picture and logout button.

Authentication System

1. You Must implement Email and password-based Authentication. This means you will have to implement the Registration and the login page.
2. Also, implement at least one extra login which could be (Facebook, github, google, etc).
3. Add a 404 page (not found page)
4. On the Login and registration page, display errors when email password authentication fails
5. On the Registration page, display errors when the password :
   a. is less than 6 characters
   b. don't have a capital letter

c. don’t have a special character
d. don’t have a numeric character

Home Page

1. Home page will have a header (simple navbar), banner (hero section), recent blog posts section, newsletter section, and footer (design in on your. Make your website beautiful using your imagination)

2. You will have to show six blogs in the recent blogs section. Show some properties from the data as you want
   a. A details button
   b. A wishlist button
3. Clicking the wishlist button will add the blog to the currently logged-in user’s wishlist.
4. In the newsletter section if a user inputs an email and submits it, then the user will receive a toast message. Message could be ”Thank you for subscribing to our newsletter” or a relevant message. You don’t need to send emails.
5. Add two extra sections on the home page with the five sections mentioned above.

   a. Tips: This is the place to shine. Add something different and unique to

   make your website look different from others. And Make them relevant to your website.

All Blogs Page

1. All blogs pages will have all the blogs added by any user.
2. Blogs can be filtered by category and add a search field to search blogs by blog title.

Tips: Use MongoDB text search, follow the doc:

https://www.mongodb.com/docs/manual/reference/operator/query/text/

3. Each blog should have a title, image, short description, category, details button and wishlist button.
4. Clicking the wishlist button will add the blog to the logged-in user's wishlist and when the user navigates to the wishlist page he/she can see the blogs they wishlisted.

   Tips: Maintain a separate collection in the same database for wishlist.

5. Clicking the details button user will navigate to the blog details page

---

Blog Details Page

1. The Blog details page will have all the information about the blog.
2. Under the blog details, there will be a textarea.
3. Here users can comment and all the comments with the comment owner's name and profile picture will be shown in the comment section. (Clue: Create a comment collection in the same database.
4. While a user comments on a blog, keep the blog's _id, user name, and user profile picture with each comment's data. So you can filter the comments by blogs _id from the backend while showing the comments of that specific blog on this details page.

5. Users cannot comment on his/her own blog. show "Can not comment on own blog" or relevant text in the comment section instead of a textarea if a blog writer is a

   current user (Clue: Compare blog email with firebase user email and conditionally render the textarea)

6. There will be a conditional update button. If the current user is a blog owner then show the update button. Clicking the update button will navigate a dynamic route where blog details can be updated. (Clue: Same as the previous clue). Description of the update route is described below.

Add Blog Page

1. Create a form which will take blog information like title, image url, category, short description, long description and a submit button.
2. category must be a dropdown element (use <select></select> tag or any package)

Update Blog Page

1. Logged-in users can update/edit the blogs they added.
2. When the user opens this page information of the editing blog will be automatically filled on the form, so that the user can only change what he actually needs to change rather than writing the whole form again
3. This page will be a private/protected route

Featured Blogs Page

1. This page will have a table including the top 10 posts.
2. Top posts will be calculated based on the word count of the long description of a specific blog. (Clue: Check the length)
3. Each table of row will include some properties from the data

---

Wishlist Page

1. In this page show all the blogs wishlisted by a specific user.
2. Each blog should have some properties as like featured blog page
3. Add a details button and a remove wishlist button
4. Clicking on remove wishlist will remove the blog from his/her wishlist

---

Challenge Requirement Guideline

1. Use server-side filter query to filter wishlist page data for currently logged-in user

2. The table on the featured blogs page will not be a normal table, implement any of the following library to create a data table, where each column of the table is sortable by clicking on the column header (You have to use one from these)
    a. Ka-table
    b. Tanstack-Table (recommended)
    c. React-data-table
    d. Mui-datatables
    e. react-table-library
3. Use the Framer Motion on the home page.
4. Implement JWT authentication on all the private routes.

Optional (But Highly Recommended)

Loading Skeleton:

1. Instead of using loading spinners use loading skeletons while data fetching a. Clue:You can explore react-loading-skeleton

Photo View:

1. On clicking the image of the blog the image will be in full-screen preview mode. a. Clue: You can explore react-photo-view

React Intersection Observer:

1. Try to implement an animation effect on a specific section if the specific section is from the viewpoint of the user. You can combine framer motion with this to implement animation (Clue: You can explore react-intersection-observer)

Component library:

1. Use any component library without daisy ui. Here are some suggestions (better to not use more than one)
    a. Chakra-ui
    b. Ant design
    c. flowbite react
    d. Rsuitejs
    e. material ui

What to Submit

Live Site Link :

Github Repository (server) :

Github Repository (client) :