assignment12_category_012

⍰ Requirement Explanation Video ⍰

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

Objective

BUILDING MANAGEMENT(Single building)

We are seeking a skilled and motivated Software Developer to join our dynamic development team. The successful candidate will be responsible for designing, developing, and implementing a robust Building Management System (BMS) that caters to both user and admin functionalities. This role requires a strong foundation in software development, problem-solving skills, and the ability to work collaboratively in a team environment.

Ensure the Following things to get 100% mark

- Include at least 20 meaningful commits on the client side & 12 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 and space The website does not express gobindo design.
- If we find your project similar to any project in your module / conceptual / assignment, you will get 0 and may miss the chance of getting any upcoming reward.

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 / Vercel
- ⍰ Logged in User must not redirect to Login on reloading any private route
- **Implement tanstack query in all the data fetching functionality (For GET method only)**

Main Tasks

Key Rules:

- Include a minimum of 20 notable GitHub commits on the client side
- Include a minimum of 12 notable GitHub 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.
- Make it responsive for all devices. You need to make it responsive for mobile, tablet, and desktop views. Make the dashboard responsive as well.
- 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 in your website.

- Show sweet alert/toast/notification for all the crud operations, successful authentication login, and sign-up. Don't use the default browser alert.
- Implement tanstack query in all the data fetching functionality (For GET method only)

Home Page

1. Navbar has a **logo + website name, Home, Apartment,** and **Login icon**(conditional). If the user is logged in, his/her profile picture should appear on the navbar replacing the login icon.

   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. Create a fancy banner with some beautiful images of the building or apartments. (Implement a feature for automatic slides on the banner.)

   Note: Do other necessary beautifications of this section.

3. Create a section named **about the building** where you need to read some details about the Building. Make this details section fancy with good **typography**.

4. You have to show the **coupons** in a fancy way in any place on your home page where users/members will be able to easily see this.

5. Create a section where you have to provide the details about your apartment's location and how to get there. You may also add an npm package for a map or an image.

   Note: Do other necessary beautifications of this section.

6. Create a footer relevant to your website. Where you should provide social links and other relevant information.

Authentication System

1. **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-
    a. Email
    b. Password
    c. Google login/ GitHub- implement any of one
    d. A link that will redirect to the Register page
2. **Register Page:** You have to use a password and email-based authentication to register. The Register page will have the following -
    e. Name
    f. Email
    g. photoURL or you can upload image through imgbb
    h. password
    i. A Link that will redirect to the login page
    ★ For password verification you need to follow this - Must have an Uppercase letter, a lowercase letter, 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

Apartment:

You have to show all the rooms here with the following information:

A. Apartment image
B. Floor no
C. Block name
D. Apartment no
E. Rent
F. Agreement button

Note: You have to store all the apartment information in the database manually (import JSON to DB).

On clicking on the **agreement button** the data will be stored in the database with the following information: (one user will be able to apply for one apartment)

A. User name(who wants to make an agreement/logged-in user)
B. User email(who wants to make an agreement/logged-in user)
C. Floor no
D. Block name
E. Apartment no
F. Rent
G. Status(pending by default)

Note: if the user is not logged in then the user will be redirected to the login page.

Note: Apply pagination at the bottom of this page. Every page will have a maximum of 6 apartment information. The rest will be on other pages following the previous rule 6 apartment information on one page.
Note: Apply a **search** functionality on top of this page where an user can search for room/apartment based on rent range. (like 1000 tk to 2000 tk rent range)
Hint: You can use two inputs for making range or you can use dropdown or whatever you want to use to make this feature functional.

User Dashboard (Private Route):

Note: **This must be a dashboard layout**

7. 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. Announcements

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

Agreement accept date and Rented apartment info such as the floor, block, room no, etc. will be 'none'. (Because normal users have no info about apartments, that's why the value will be 'none' for these fields.)

9. **Announcements:**
   You have to show all the announcements on this page announced by the owner.

Member Dashboard (Private Route):

Note: **This must be a dashboard layout**

10. 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. Make payment
    C. Payment History
    D. Announcements

11. **My Profile:**
    This page will have the user's name, image, email, Agreement accept date, and Rented apartment info (floor, block, room no, etc.)

12. **Make payment:**
    On this page, there will be a form containing some fields named member email(read-only), floor(read-only), block name(read-only), apartment no/room no(read-only), rent(read-only), month(which month's rent you want to pay) and a submit/pay button. (these are member's agreement apartment's/room's information)
    On clicking on the pay button member will be redirected to a page where he/she will be able to make payment. You have to add a field and a button where members will be able to apply coupons.
    On clicking the button if the coupon is valid then the rent will be reduced by the given percentage on the coupon. **(see the challenge requirement-4 for rest)**

13. **Payment history:**
    This page will have a table with the payment details of the logged-in user.

14. **Announcements:**
    You have to show all the announcements on this page announced by the owner.

Admin Dashboard (Private Route):

Note: **This must be a dashboard layout**

15. 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(see bonus tasks)
    B. Manage Members
    C. Make Announcement
    D. Agreement Requests
    E. Manage Coupons

16. **Manage Members:**
    Show all the Members in a tabular form where each row will have:
    - User name
    - User email
    - Remove button

    On clicking on the **remove** button the role of that member will be changed to user and he/she will lose access to the member dashboard. he/she will be able to access the user dashboard.

17. **Make announcement:**
    This page will have a form with the following fields:
    - Title
    - Description

What announcement you will make is entirely up to you but make sure to keep it relevant.

18. **Agreement requests:**
    All agreement request information will be shown here with the following information:
    a. User name
    b. User email
    c. Floor no
    d. Block name
    e. Room no
    f. Rent
    g. Agreement request date
    h. Accept button
    i. Reject button

On clicking the accept button the **status** will be changed to **checked** and the **user's role** will be changed to a **member**. On clicking the reject button the **status** will be changed to **checked** and the **user's role** will remain the same.
Note: After the operation, the data will be removed from here.

19. **Manage coupons:**
    Show all the coupons in **tabular** form added to the database by the owner/admin.

    Anywhere on this page, there will be an **add button**. On clicking that there will be a **modal** with some fields named **coupon code, discount percentage, coupon description, and submit button**.
    On clicking the submit button the coupon data will be stored in the database.

Challenge Tasks

1. **Admin Profile:**
   This page will have the following pieces of information:
   a. Admin's name
   b. Image
   c. Email
   d. Total number of rooms in the database
   e. Percentage of available rooms in the database
   f. Percentage of agreement/unavailable rooms in the database
   g. Number of users in the database
   h. Number of members in the database

2. Implement JWT on login (Email/Password and social) and store the token(you have to use localstorage to store the token, cookies won't be allowed).

3. The admin/owner will be able to change the availability of a coupon.

4. On successful payment, the data**(given in requirement 12)** will be stored in the database and a successful message will be shown.

Optional Tasks

You can implement any of the tasks given below:

1. Create a feature where if a member's rent is due then he/she will get a notice from the owner.
   After 3 consecutive notices, the member will be removed from the apartment making his/her role member to the user.
2. Implement axios interceptor.
3. Use any animation package like Framer Motion or others.

What to Submit

Copy and paste the following text and then add the required links and then submit the assignment:

- Admin email:
- Admin password:
- Front-end Live Site Link:
- Client Side Github Repository Link:
- Server Side Github Repository Link: