Assignment 11 Requirements

Assignment_ID: assignment_category_0009

Here is video for more clarification -**link**

A company is looking for a full-stack web developer capable of creating a simple full-stack website similar to online marketplaces.

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

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

    - Dragon news
    - Espresso Emporium
    - Genius Car
    - Career Hub Website, any conceptual session projects, or projects showed in our course
    - Any of your previous assignments 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.)

3. You should have a navbar with the name+logo,Home,Add job,My posted jobs,My Bids,Bid Requests,user profile picture and user name and Register/login
4. **Login & Registration systems:** On the Registration and Login pages, display relevant error messages when necessary

   Login Page: When a user clicks on the login button, they will be redirected to the login page having the following:

   - Email/Password
   - Google Sign-in
   - A link that will redirect to the registration page

   Registration Page:The Registration page will have the Email/Password form having the following fields:

   - Name
   - Email
   - Password
   - PhotoURL

   Note: Do not enforce the email verification method and forget & reset password method, as it will inconvenience the examiner. If you want, you can add after receiving the assignment result

5. **Home page**
   - **Banner**
     - Banner section will have a banner/carousel (images should be relevant to the website)
   - **Browse By Category**
     - The website will feature a tab-based system for browsing jobs, representing the categories of web development, digital marketing, and graphics design. Each tab will display a minimum of four cards.
     - These job data will come from mongodb database. Users will add job data from the Add Job page. [See No. 7]
     - Implement tabs using this NPM package: React-tabs

Note: [On this card, you will have to show the following information:

- Job title
- Deadline
- Price range
- Short description
- Bid now Button

- Added extra two sections relevant to the website .

6. **Job details page:**
   After clicking on the bid now button, he/ she will be redirected to the job Details route (/jobs/:id containing the information (name, Deadline,price Range,description and **place your bid form** section on the bottom of the card.

   Note: The “place your bid” form section will have the following input fields:

   - Price(your bidding amount)
   - Deadline
   - Email(read-only-get it from user context.) (Who is bidding)
   - Buyer Email(read-only) (The job owner who has posted this job.)
   - Bid on the project button (This button will be disabled for the owner who posted this job)
   -

   Note: When the '**Bid on the project**' button is clicked, data will be stored in a MongoDB database. Once the data is saved, show a toast [don’t use javascript alert] and after that user will be redirected to the **My Bids** page. [See No. 9 for detail information about **My Bids** page]

   Don’t let the employer bid on his own job.

7. **Add jobs:** Create an Add Jobs page where there will be a form having the following fields

   - Email of the employer(read-only)
   - Job title
   - Deadline
   - Description
   - Category-these are the names of tab options of the homepage(implement a dropdown for selecting category)
   - Minimum price
   - Maximum price

- Add job button

Note: When the '**Add Job**' button is clicked, data will be stored in a MongoDB database. Once the data is saved, show a toast [don't use javascript alert] and after that user will be redirected to the **My Posted Jobs** page.[See No. 8]

8. **My posted jobs:** If a user logs in, they will see the My posted jobs page which will show all the jobs information (he/she) added from the **Add jobs** page. Each card will have an **update** and **delete** button.

    - Update action- If they click the update button, they can update the job information

        - email(not editable)
        - Job title
        - Deadline
        - Description
        - Category
        - Minimum price
        - Maximum price
        - Update button

    Note: you can show the update form in a modal or another route. When the '**update button**' is clicked, data will be updated . Once the data is updated, show a toast [don't javascript alert]

    - Delete action- After clicking the delete button, the job will be removed. Before the delete, ask for a delete confirmation.

    Note: If a user logs in they will only see the jobs they have added. The user cannot see the jobs other users added.

    ****Don't Let a user Update/Delete other employers job post.****

9. **My bids:** Create a My bids page where you will show all the bid information (he/she) added from the job details page in the tabular form. Each row of the table will have the following information:

- Job title
- Email
- Deadline
- Status
- Complete button(conditional)

Note: initially status is **pending**.If the job owner rejects the bid then status will be **canceled**. Or if the bid is accepted by the job owner (who posted this job) status is **in progress.** If the status is **in progress** the complete button is enabled. After clicking the **complete button**, status is **complete** and the complete button will disappear again.

10. **Bid Requests:** job owner will see all the bid requests made by the users on their posted jobs in tabular format in this page. Job owner can manage (accept/reject) job status on this page. Each row of the table will have the following information:

    - Job title
    - Email (who bid the job)
    - Deadline
    - Price
    - status
    - Accept button
    - Reject button

    Note: Initially accept button and reject button will be enabled. after clicking the Reject button the status will be changed to **rejected** and both accept button and the reject button will Status will be updated to **in progress** and both accept button and reject button will disappear In the place of these buttons a progress bar will be visible containing the current status of the job .(in progress-completed). Making the progress bar dynamic is a optional task, but we highly encourage you to give it a try by using this package: react-step progress-bar

11. **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.

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

13. **Private Route:**

- My bids page
- Add job page
- My posted jobs
- Bid Request page
- Job detail page

Bonus Requirement

1. Commits & readme:

    - Minimum 18 meaningful git commits on the client-side repository.
    - Minimum 10 meaningful commits on the server-side repository.
    - Create a readme for client-side and write about your project (at least 5 bullet points).

    **** Remember to add your client-side live link to your website here.****

2. Reload: If you reload the protected/private routes (after login), this page will not redirect the user to the login page. Instead, it will keep the logged-in user on the protected route.

3. 3. Make the Homepage of your website mobile, tablet & desktop responsive.

4. On the My bids page, you must implement a sorting system to sort the jobs based on the status by ascending order. (Explore MongoDB sorting operation & implement it on the backend). The design of the sorting system depends on you.

5. Give your website a name. The website title will be changed according to the route you are clicking. Suppose your website name is PHero. Then, on the 'All Products' route, your website title will be 'PHero | All Products'. Don't forget to change the favicon.

6. Upon login, you will create a jwt token and store it on the client-side and you will send the token with the call and verify the user. Implementing 401 and 403 is optional. Ensure you have implemented jwt token and 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.

Optional (But Highly Recommended)

Implement any two tasks from the following optional list:

1. 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.
2. Explore and implement any of the animations from the **Framer Motion**
3. Explore and implement **Tanstack query** mutations in your api.
4. Add one extra feature of your own. This will help you in the future to differentiate your project from other

Make the deadline strict:

1. When adding a job, take the deadline input as a valid date.
2. If a job's deadline is crossed (compare using js Date.now()), no one can place a bid on it. It means the "Bid on this project" button will be disabled based on the deadline.

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

What to submit:

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