**assignment\_category\_07**

Dear Candidates,

Are you a talented Full Stack Developer passionate about creating impactful web applications? Do you enjoy working with modern technologies like React, Firebase, Node.js, and MongoDB? We’re building a Community Food Sharing and Surplus Reduction Platform and need a developer who can bring innovative ideas to life. You’ll design and maintain a dynamic, responsive platform while implementing robust security and user-friendly features.

**Food Sharing Website**

You have to build a food sharing website where you have to implement CRUD (adding food,getting food, update food and deleting food) operations on several places. Besides that, users can also request for available food and manage requested foods.

#### **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: Home,,,.
   1. Home
   2. Available Foods
   3. Add Food (Private)
   4. Manage My Foods(Private)
   5. My Food Request(Private)
   6. Login
   7. Signup

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. **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-
   1. Email
   2. Password
   3. Google login/ GitHub- implement any of one
   4. 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.**

1. **Register Page:** You have to use a password and email-based authentication to register. The Register page will have the following -
   1. Name
   2. Email
   3. photoURL
   4. password
   5. 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.**

#### **Home Page**

**Banner/Slider:**  Add a beautiful banner/ slider . Try to make it catchy and impressive.

**Featured Foods :** In this section, you will have to show 6 Food items.

The information you want to show is up to you.

**Note :** Featured Foods represent the highest quantity of food options available.

**A Show All button** that will redirect you to the Available Foods page.

**Extra Section:** Add 2 relevant extra sections in the homepage in addition to the

Featured Section. Try to make them attractive.

Note: When a user is not logged in and if he/ she clicks on the View Details button,

redirect him/ her to the login page. Without a login, you can not visit the single food details page.

#### **Add a Food (PRIVATE)**

Create an Add Food page where there will be a form having the following fields:

* Food Name
* Food Image
* Food Quantity
* Pickup Location
* Expired Date/Time
* Additional Notes
* Add Button
* Donator Image , Name, & email (From logged in user)
* Food Status (By default keep it ”available”)

Clicking on the add button the data will be saved on a collection (food collection) and the added food will be shown in the Available Food Page.

#### 

#### **Available Foods Page**

Hint: Show the foods where the food status is available.

**Sorting Section:**

You have to implement Sorting functionality by the Food Expire Date.  
[documentation](https://www.mongodb.com/docs/manual/reference/method/cursor.sort/)

**Foods Section:**

You have to show all the available foods in this section in card format.The information you want to show is up to you. There will be a View Details Button. On clicking the Details button navigate to the food details page.

Note: If you apply any sort or search parameters then show searched Foods/Sorted Foods instead of all foods.

On clicking the View Details button will redirect to the Food details page.

#### **Single Food details**

After clicking on the View Details button, he/ she will be redirected to the Food

Details route ( /food/:id ).Show all the information of food. There will be a Request button in details page

On clicking Request Button button a modal will open with following input field

* + Food Name( Not editable )
  + Food Image ( Not editable )
  + Food Id (Not editable)
  + Food Donator email ( Not editable )
  + Food Donator Name (not editable)
  + User email ( LoggedIn user , Not editable )
  + Request Date(current time not editable)
  + Pickup Location(not editable)
  + Expire Date(not editable)
  + **Additional Notes** (editable)
  + Request Button

On clicking the Request Button, the food will be removed from the available foods and added to My request foods.

Hint: The food status will be changed to “requested”

#### **Manage My Foods (PRIVATE)**

This page will be a private route. If a user logs in, they will see the Manage Food page, which will show all the foods in tabular format they have added from the Add Food page.

Each Row will have an updateand delete button

**Update Action** - If they click the update button, they can update the food

information

Note: you can show the update form in a modal or another route.

**Delete Action** - If they click the delete button, the food will be removed.

Before the delete, ask for a delete confirmation.

**Note**: If a user logs in they will only see the foods they have added. The user

cannot see the foods other users added.

#### **My Food Request (PRIVATE)**

The page will contain All Food Requests made by the logged-in user. Show the data in a tabular format or design as you wish but you must show the information which you have taken while doing a food request. Some of the info examples are given below.

* Donar Name
* Pickup Location
* Expire Date
* Request Date

#### **Challenge Requirement Guideline**

**Layout:**

Initially there will be a three column layout to show the foods in the **Available Foods Page**(see Available Foods Page requirement) page. On clicking the change layout button, the layout will be changed to a two-column layout.

(Hint : use state to toggle between 3 grid column and 2 grid column)

**TanStack Query:**

Implement Tanstack query for fetching at least 1 API. utilize the mutation feature of tanstack query.

**Search Section:**

You have to implement search functionality by the Food name on the available food page.

**Do Some Security with JWT:**

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. Explore and implement any of the animations from the Framer Motion
2. Add one extra feature of your own. This will help you in the future to differentiate your project from others.
3. Use Axios Custom Hook. [AxiosSecure]
4. Try some other Tailwind CSS library for design like Shadcn , Mamba UI, Preline,Chakra UI

#### **What to Submit**

Live Site Link :

Github Repository ( server ) :

Github Repository ( client ) :