

CSE470 : SOFTWARE ENGINEERING

# Project Title : Product Inventory Website

**Section 04**

**Group 08: CLUELESS**

**Summer 2023**

**SUBMITTED BY**

| **Name** | **ID** |
| --- | --- |
| Mollah Md Saif | 20101416 |
| Nourin Siddique Ananna | 20301012 |
| Mazharul Islam Rakib | 20101408 |
| Mahir Ahmed Niloy | 19101114 |

**SUBMITTED TO**

MD. SHAZID BIN MAHBUB

LECTURER, BRAC UNIVERSITY

**SUBMISSION DATE:** 28/08/2023

# Table of Contents

| **Section No.** | **Content** | **Page No** |
| --- | --- | --- |
| **1** | Introduction | 2 |
| **2** | Functional Requirements | 2 |
| **3** | User Manual | 3 |
| **4** | Frontend Development | 8 |
| **5** | Backend Development | 9 |
| **6** | Technology (Framework, Languages) | 12 |
| **7** | Github Repo Link | 12 |
| **8** | Individual Contribution | 12 |

# Introduction

Product Inventory Website is a platform where users can record the inventory that they have in their homes solely for personal use. The user can track their food in the pantry, fridge and other staple pantry goods. They track anything starting from clothes, books to medicine etc.

## Background

In the age of mass production and consumerism, people tend to forget the things they have at home. They tend to overspend and have multiple items of the same product or food. This creates waste since many get thrown away in the landfill. This problem is severe with food, people forget what they have in their home. As a result, excess food gets thrown away. With this website, we want to install good buying habits in users so that they spend wisely and create less waste.

## About this project

With the technology of MongoDB, Node.js and React, we have created a website where the users can log in and record their inventory from different categories. They can set reminders and can track their usage as they use that item. They also have the ability to customise reminders and lists for the product they are tracking. They have to create an account to log in and can keep their record saved.

# Functional Requirements

The Functional Requirements of the project is as follows:

1. Create new account
2. User login
3. A dashboard with calendar, remainders and inventory tracking
4. A statistics page with user tracking, money spending, amount in different categories
5. A product page for product tracking and filtering with different categories and price; it will have a start and end date, price, category and amount
6. Ability to export, import data from the logged in data

# User Manual

Our web interface only has a user side. Since this is a simple inventory tracking for personal use, admin authentication and tracking is not required.

## Login and Registration

The user has to create an account with a valid Name, Phone Number, Email ID and password. The user can then log in with the registered account to start using the interface.

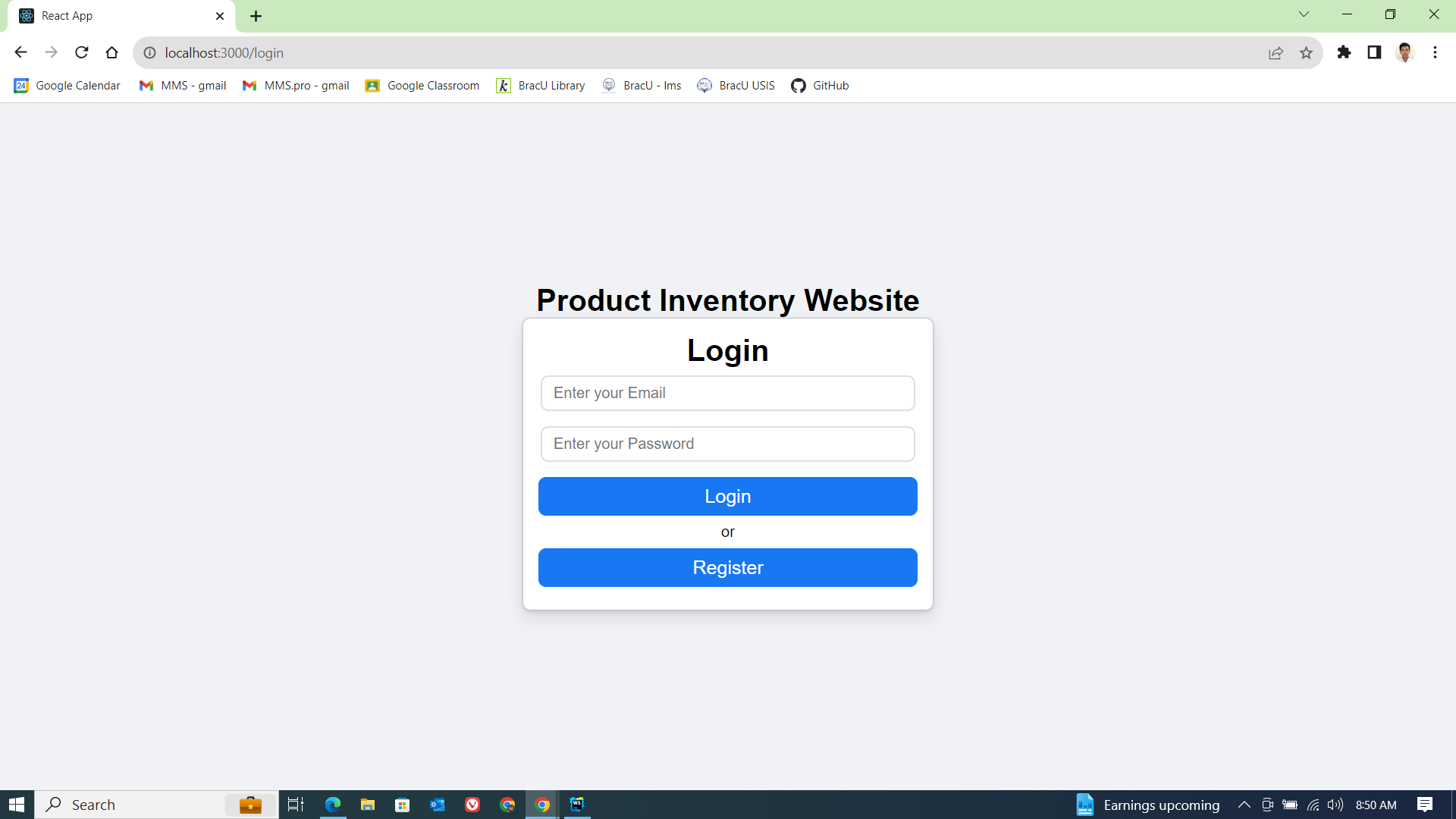


Fig1: The login interface of the website

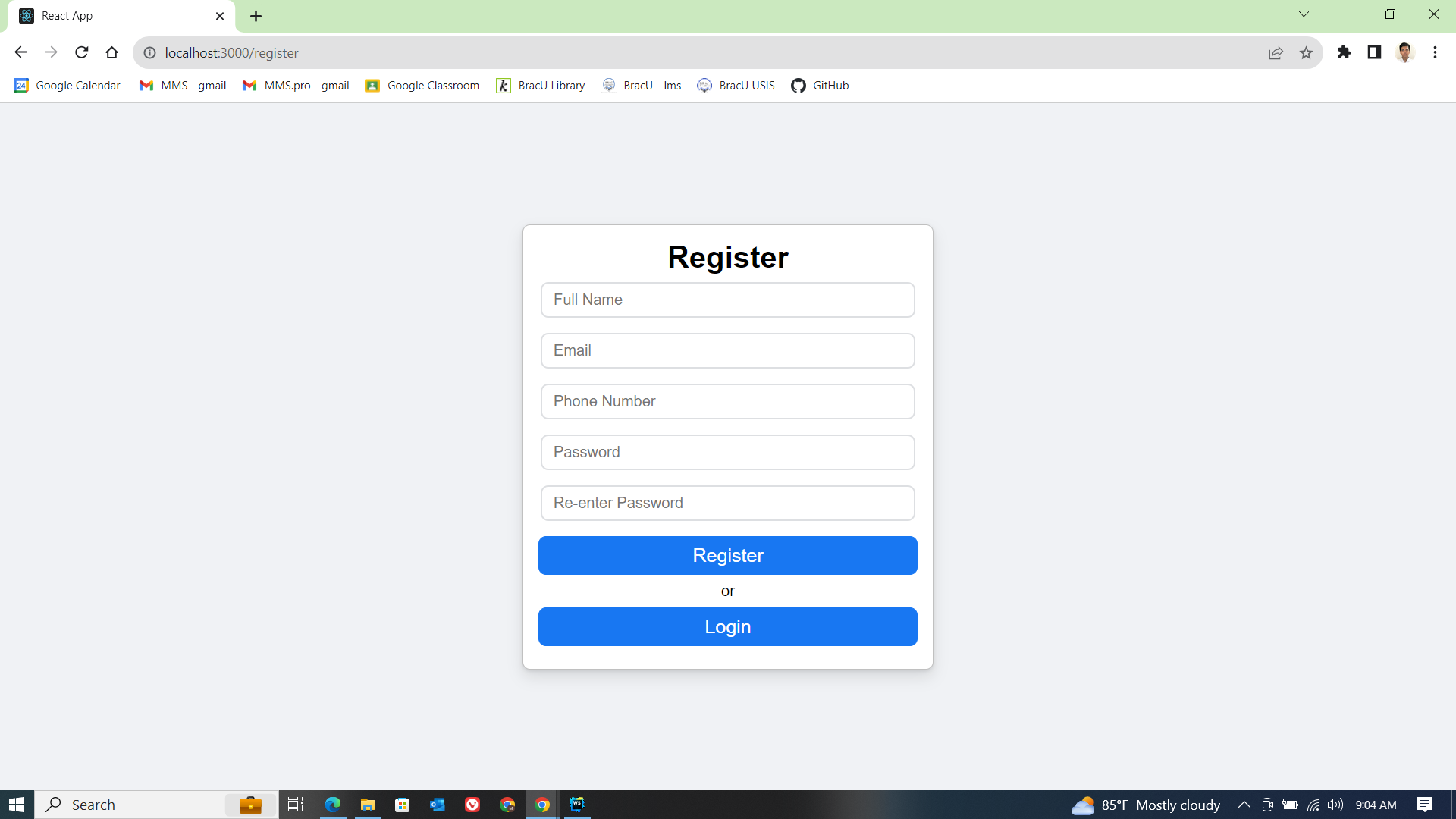


Fig2: The Registration interface of the website

## Dashboard

### ***Calendar****:* Once the user is logged in, they see Figure 3 initially. After they scroll down, they get to see figures 4 and 5. The dashboard provides a holistic view of the website. Here we have a calendar to see which day it is. We can add a note here for the user to remind themselves of anything they need to buy or donate.

### ***Grocery List:*** We have a grocery list where the user adds items they need to shop for when they next go to the shop. This helps the user not to overspend and to shop responsibly. Using the add grocery button, we can add items to shop for.

### ***Expired Product list:*** The expired product list consists of everything which is going to expire or finish. This acts as a reminder for the user to stock up on the product before it runs out.

* ***Remainder list:***  The remainder list consists of all the products the user wants to be reminded of or any other remainder such as when to buy something etc.
* The dashboard also displays a sidebar consisting of inbox, dashboard, product page and data. To display the statistics, we press the *view statistics* button. We also have a logout button to help the user logout.

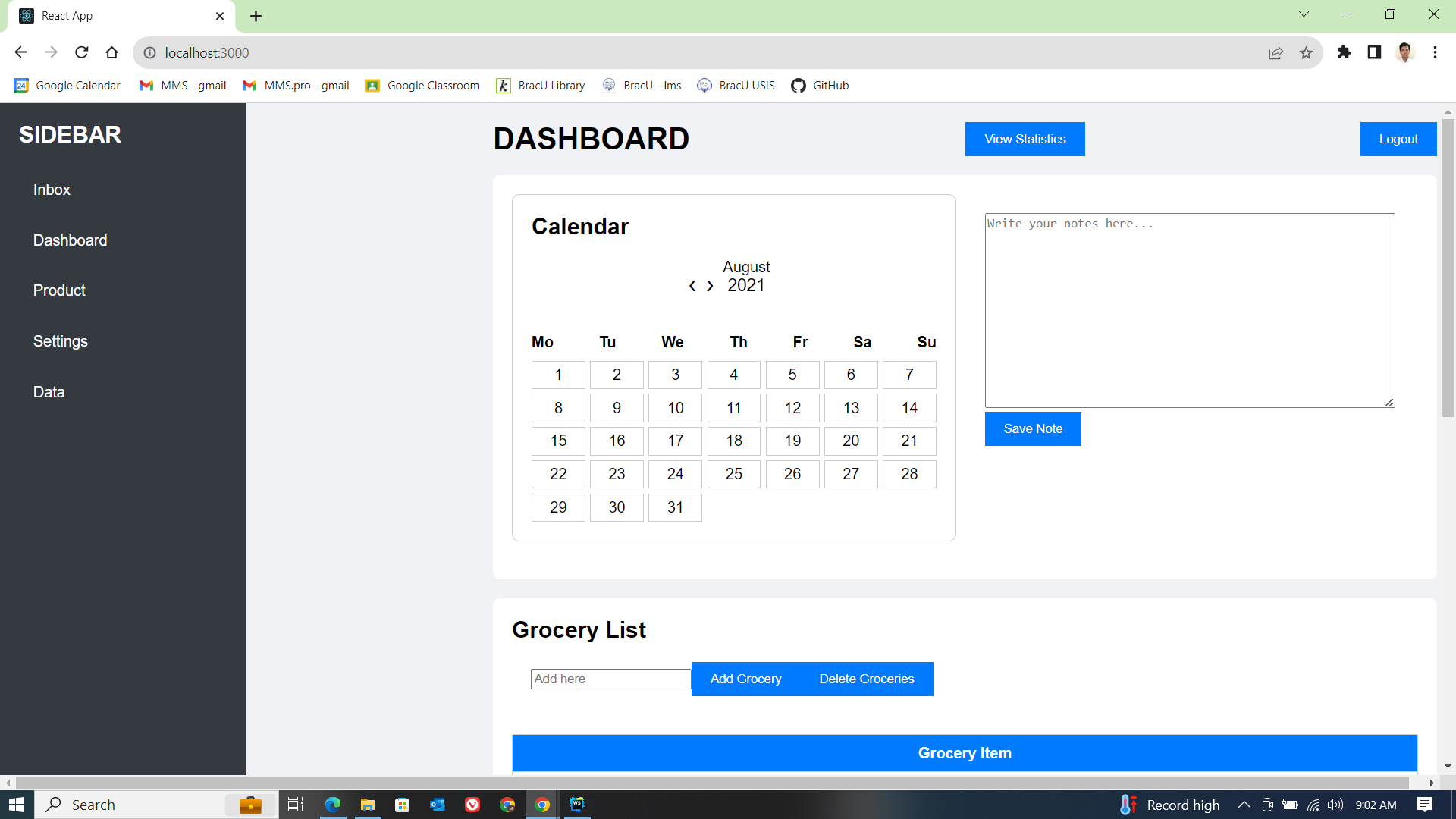


Fig3: The dashboard-1

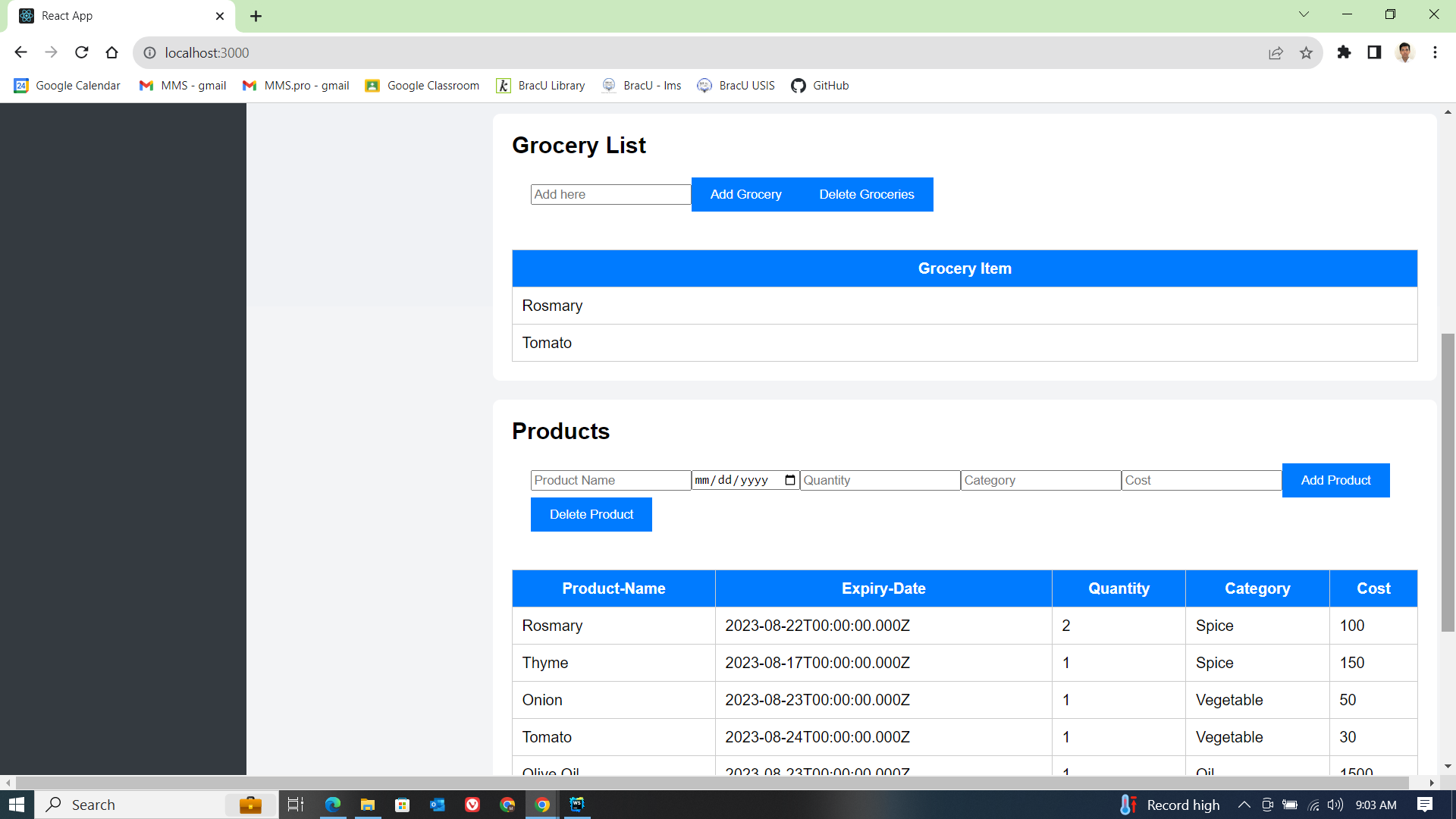


Fig4: The dashboard-2

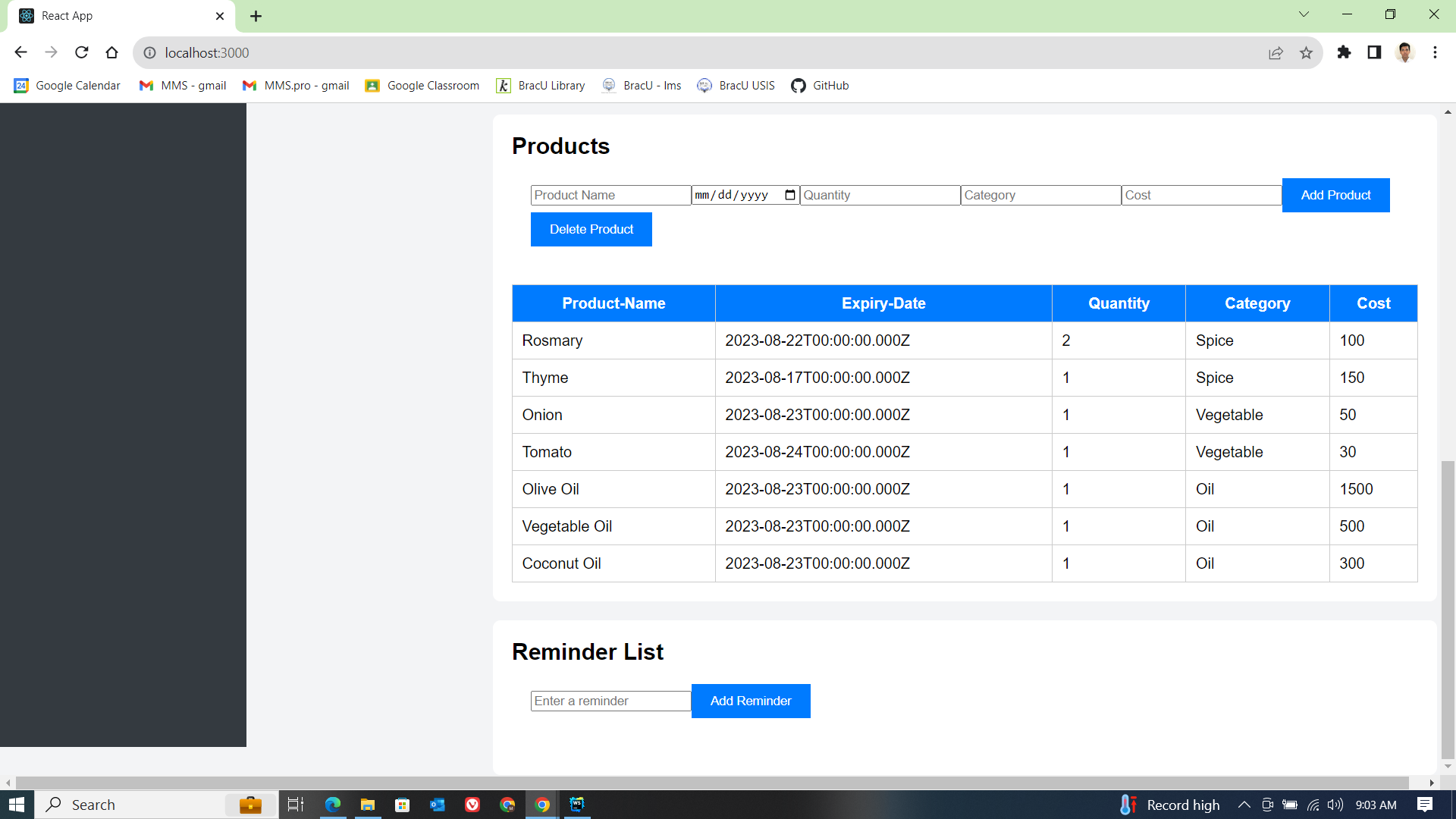


Fig5: The dashboard-3

## Product Page

The product shows all the products in a table and filters the data by category. By typing the category in the Filter Category prompt we will get filtered products. The categories are product name, expiry date, quantity, product category and cost.

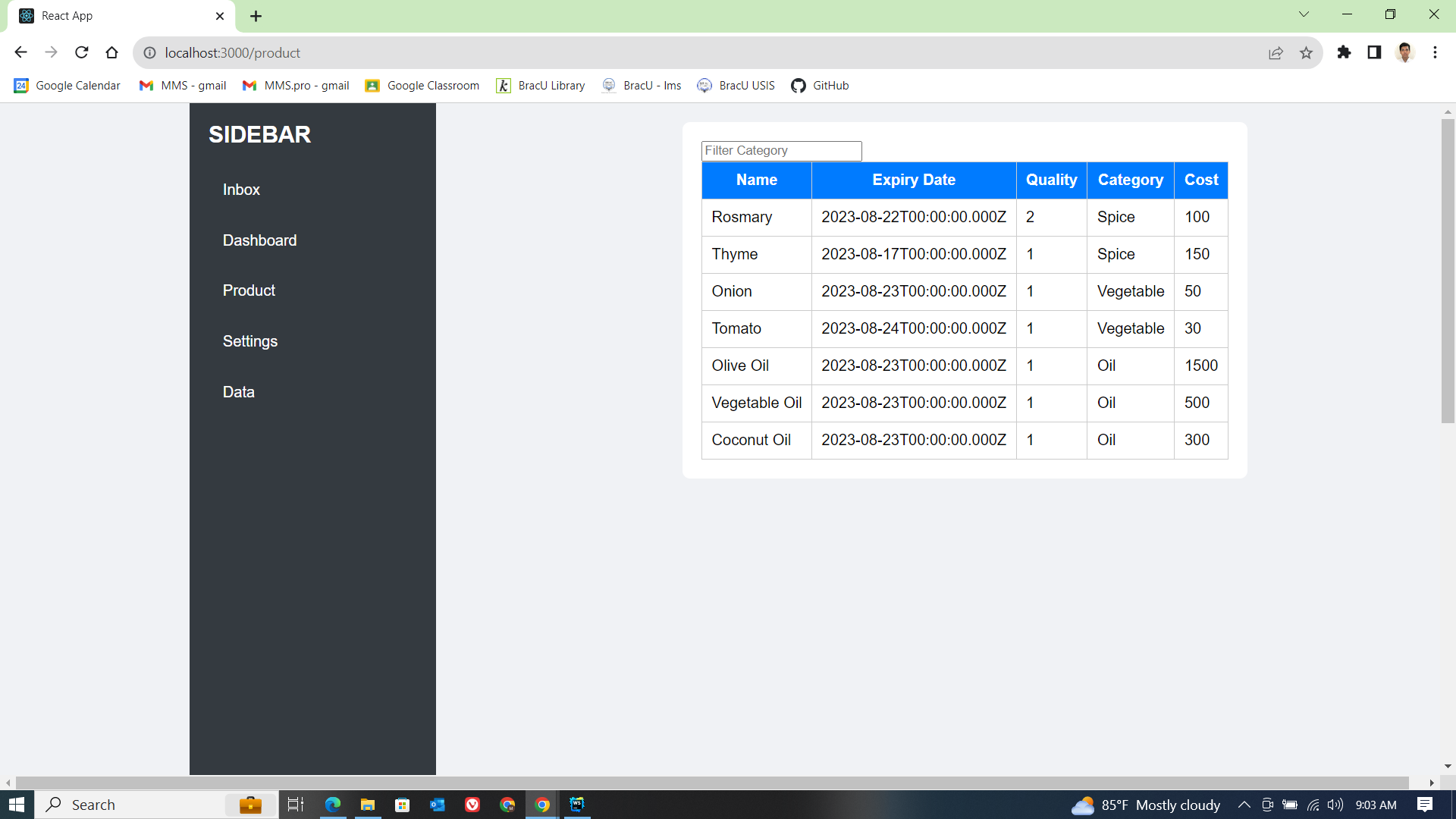


Fig6: The Product Page

We add a product by giving the name, date, quantity, category, and cost of the product and press the Add Product button.

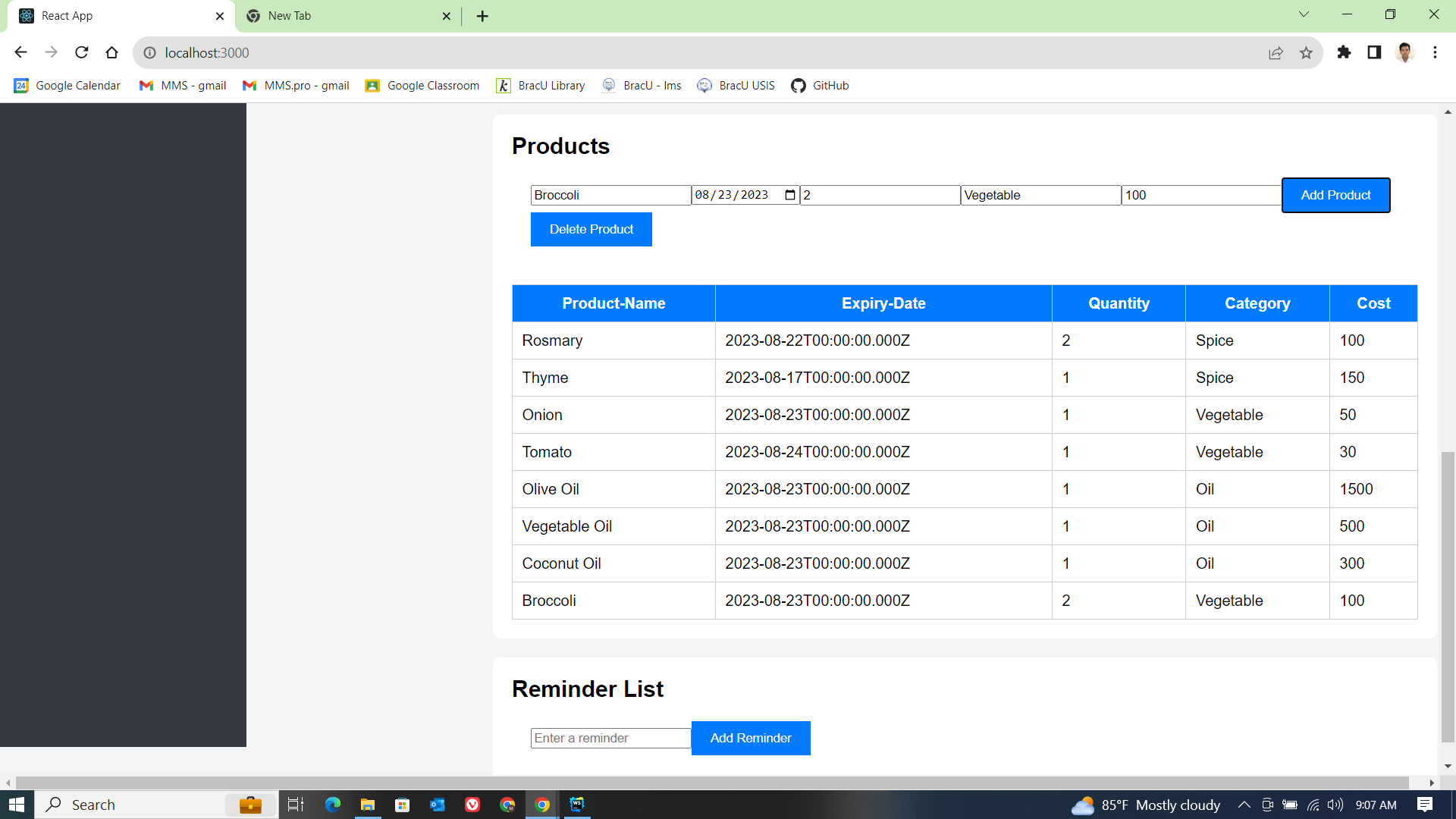


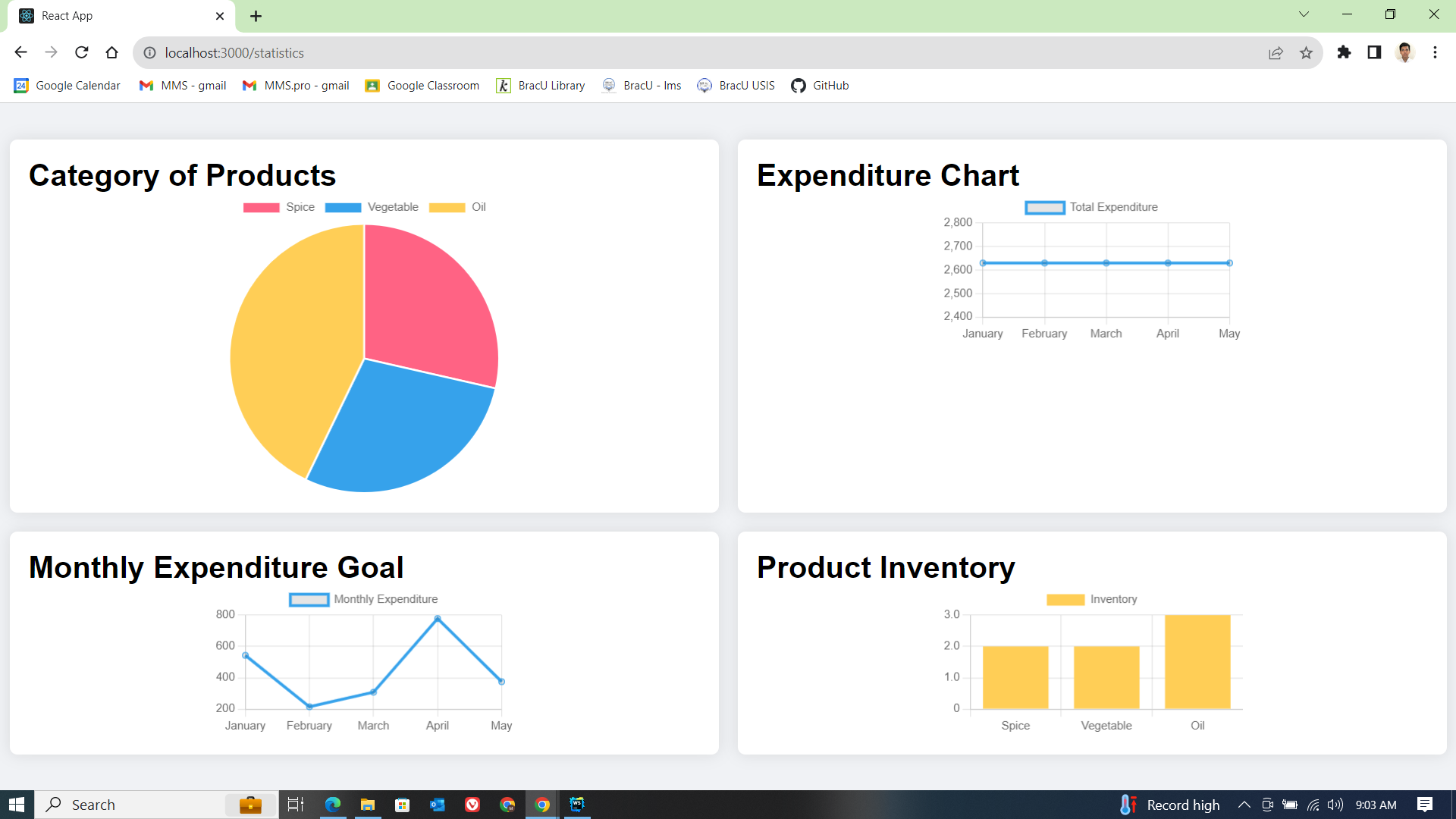
Fig7: The Adding Broccoli as Product in Product Page

## 

## 

## Statistics

The Statistics page shows a summary of products in four charts. First graph shows the distribution of the category of products in a pie chart. Second graph shows an expenditure graph showing the expenditure by each month. Thirdly, we added a monthly expenditure goal in a line graph. Lastly, there is a product inventory bar chart showing each category and its quantities.

 Fig8: The Statistics Page

## Data Manipulation

The data manipulation page has two functions. They are:

* **Data Import:** For importing products, we need to choose a json file containing the products and then we need to click the upload file button which will upload the data into the database.
* **Data Export:** By clicking the Download Data button it fetches all product data from the database and downloads in a json file.

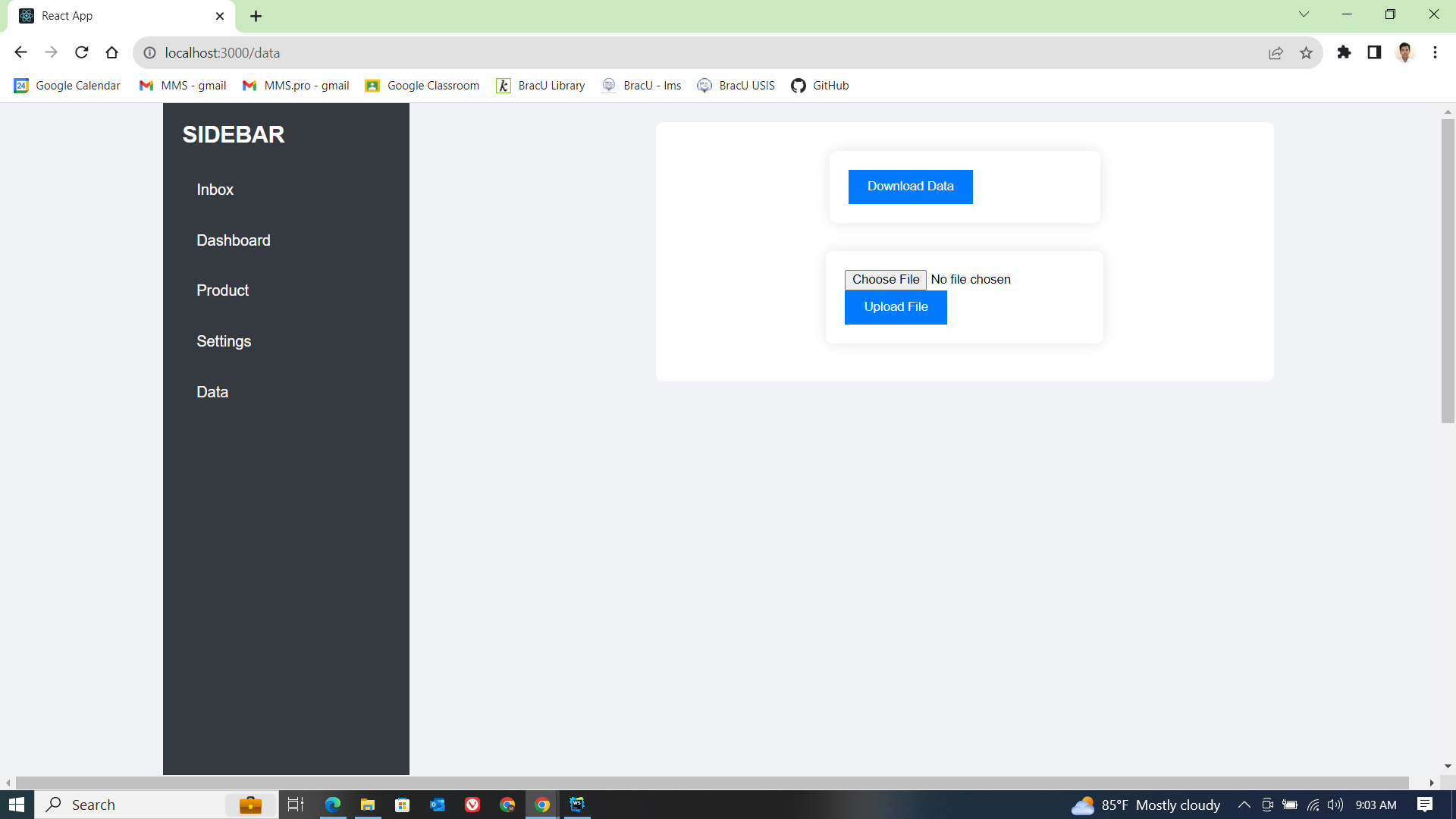


Fig9: The Data Manipulation Page

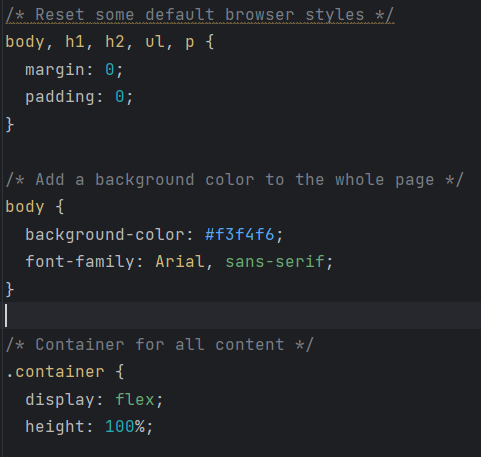
# 

# Front End Development

In the front-end section of our project we used HTML, CSS and Javascript programming language. For a structured and responsive website, we built our website on React and Node.js framework. For keeping a clean architecture and minimal configuration React Framework was chosen. Also for hosting front-end we used Node.js for its MVC architecture. We have rendered HTML from javascript files. A code snippet of homepage.js:



For styling purposes we kept separate CSS files for each module. For example a style.css for homepage:



For each Module there is a page that is rendered from a javascript file. The javascript helps scripting the webpage and communicates with the database.

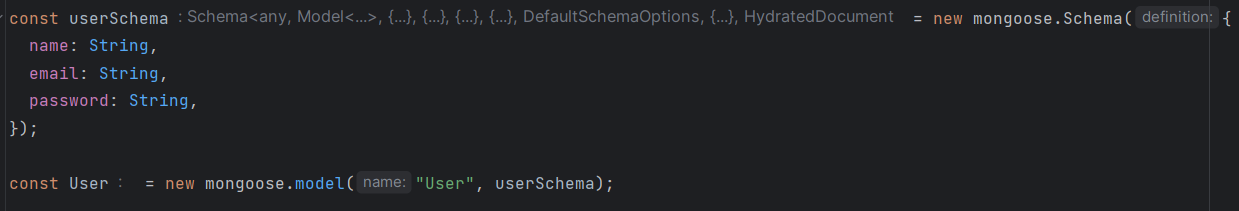
# Back-end Development

Javascript is used for our back-end programming language. For keeping creating Routing, Middleware, and API easily we used Express Framework. We have used MongoDB for our database. MongoDB allowed us to host the database in the cloud and connect with the Express server very conveniently. The code snippet of connection of MongoDB with back-end:



## User Model

User Model is a mongoose data model which instantiates a user object and stores the user data inside a mongoose object. The code snippet is:

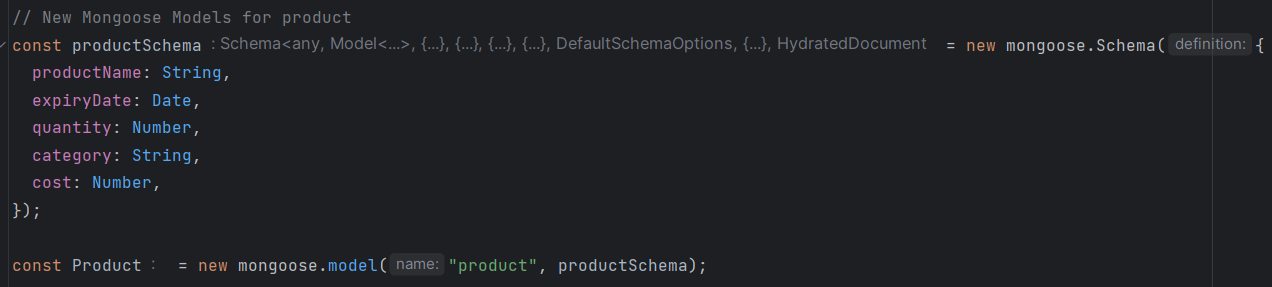


Using the user object we use two API functions. They are login and register. The code snippet is:

| Function | Description |
| --- | --- |
| login(); | Get the login credential and check if the user exists. |
| register(); | Post the login credential in the database |

## Product Model

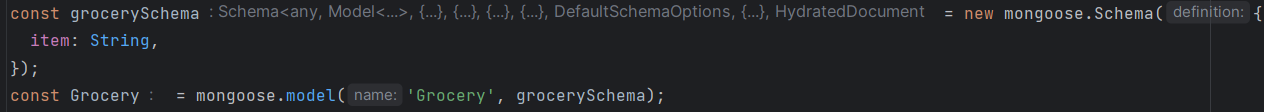
Product Model is a mongoose data model which instantiates a product object and stores the user data inside a mongoose object. The code snippet is:

Using the product object we use three API functions are

| Function | Description |
| --- | --- |
| addProduct(); | Adds a product to the products to the database. |
| getProducts(); | Retrieve all the products from the database. |
| getCategories(); | Retrieve all the categories from the database. |
| deleteProduct(); | Delete a product item from the database. |

## Grocery Model

Grocery Model is a mongoose data model which instantiates a Grocery object and stores the user data inside a mongoose object. The code snippet is:

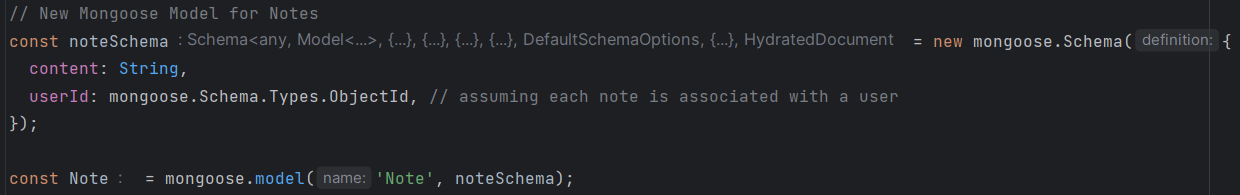


Using the grocery object we use three API functions are:

| Function | Description |
| --- | --- |
| addGrocery(); | Adds a product to the grocery item to the database. |
| getGroceries(); | Retrieve all the grocery items from the database. |
| deleteGrocery(); | Delete a grocery item from the database. |

## Note Model

Note Model is a mongoose data model which instantiates a note object and stores the user data inside a mongoose object. The code snippet is:



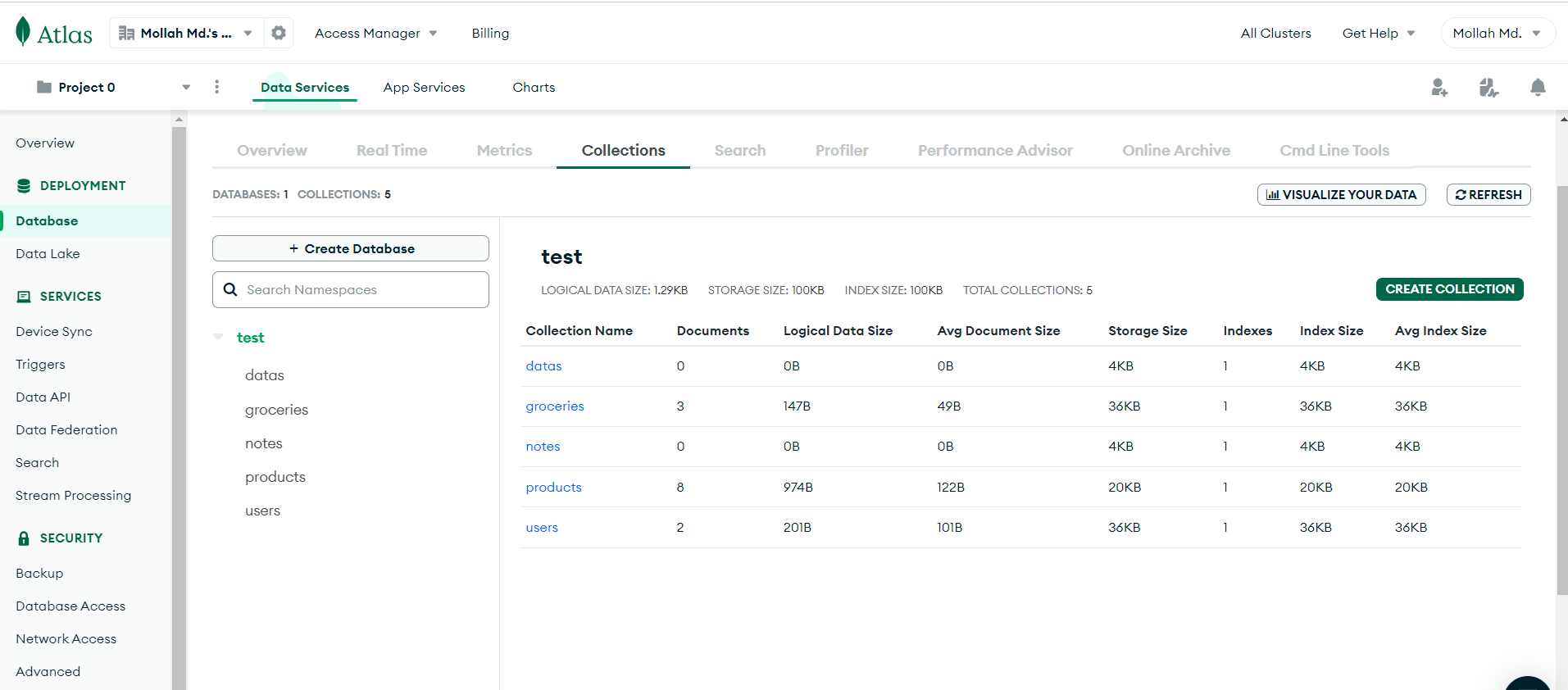
Using the note object we use three API functions are:

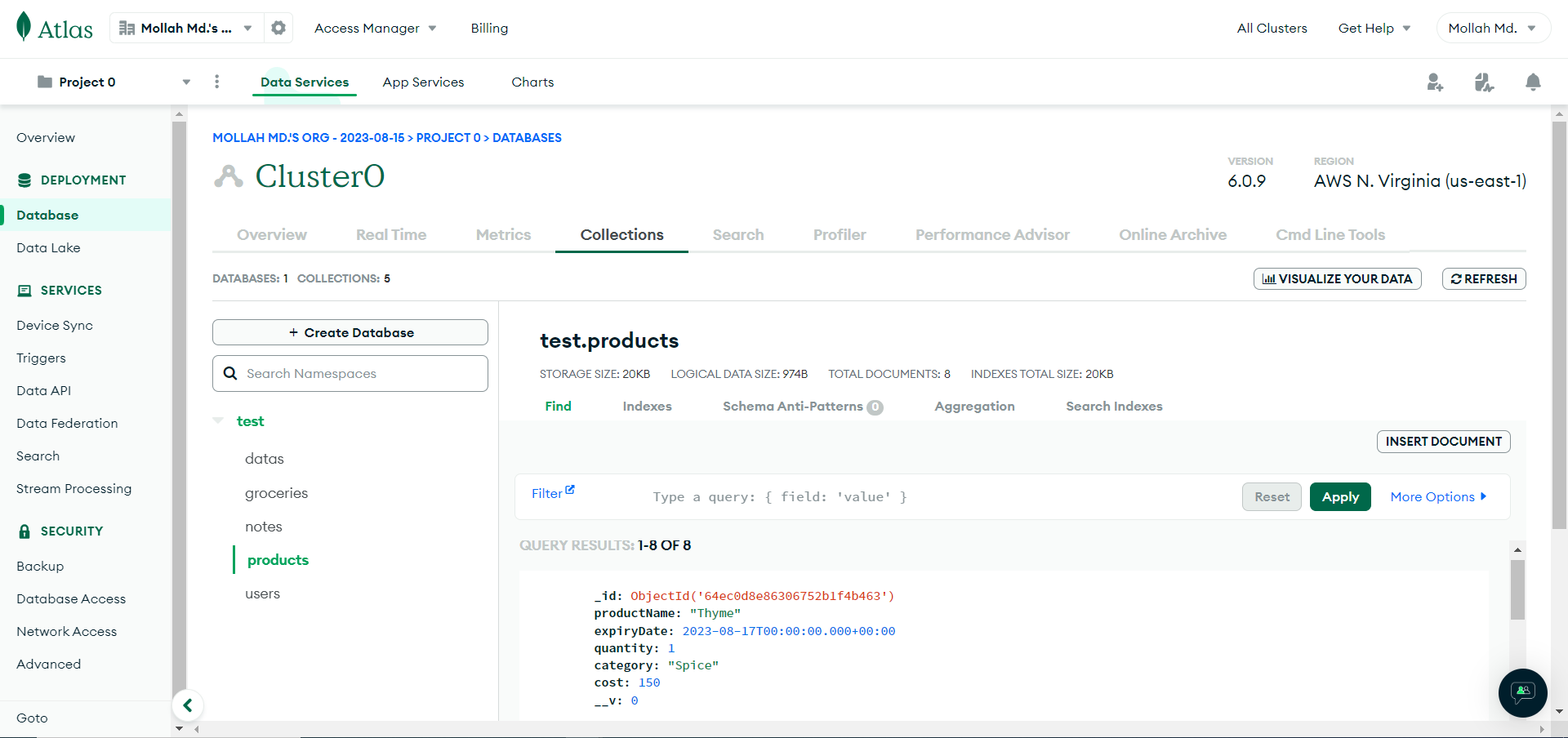
| Function | Description |
| --- | --- |
| saveNote(); | Adds a note item to the database. |
| getNotes(); | Retrieve all the notes items from the database. |

For importing and exporting data from the database we use following two API functions:

| Function | Description |
| --- | --- |
| app.post('/api/upload') | Import json data to the database |
| app.get('/api/items') | Export all data from the database and return |

### *MongoDB Database*





# 

# 

# Technology (Framework, Languages)

| Type | Useage | Name |
| --- | --- | --- |
| Language | Front-end | HTML |
| Language | Front-end | CSS |
| Language | Front-end and Back-end | Javascript |
| Framework | Front-end and Back-end | Node.js |
| Framework | Back-end | Express.js |
| Framework | Back-end | MongoDB |

# 

## Github Repository Link

https://github.com/Rakib3103/Product-Inventory-Website.git

# Individual Contribution

# 

| ID | Name | Contribution |
| --- | --- | --- |
| 20301012 | Nourin Siddique Ananna | Report |
| 20101416 | Mollah Md Saif | Front-end and Backend of Product Page and Data Manipulation Page |
| 20101408 | Mazharul Islam Rakib | Front-end and All Backend of Login, Register, Homepage, Statistics Page, Inbox, Website Routing |
| 19101114 | Mahir Ahmed Niloy | Front-end of Register and Login |

# 

# 

# 