BRAC UNIVERSITY

Inspiring Excellence

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

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.

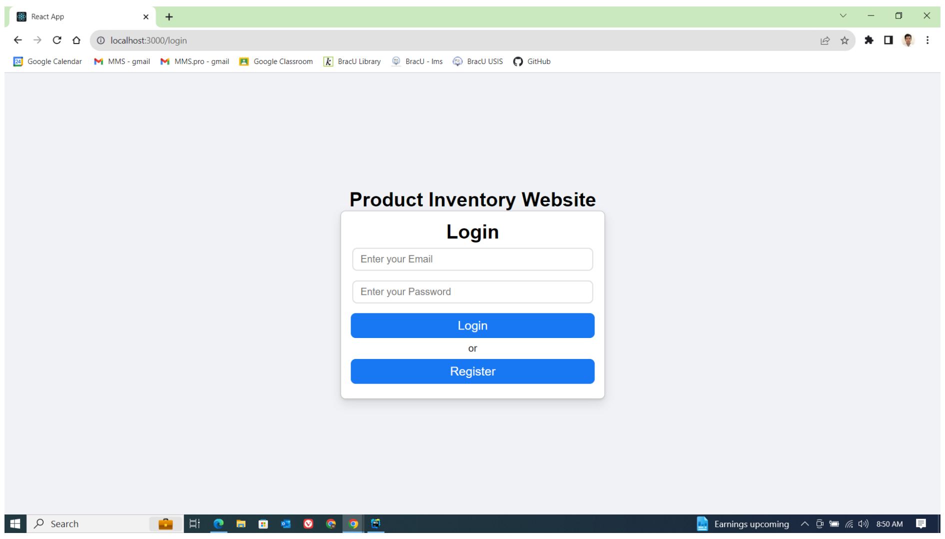


Fig1: The login interface of the website

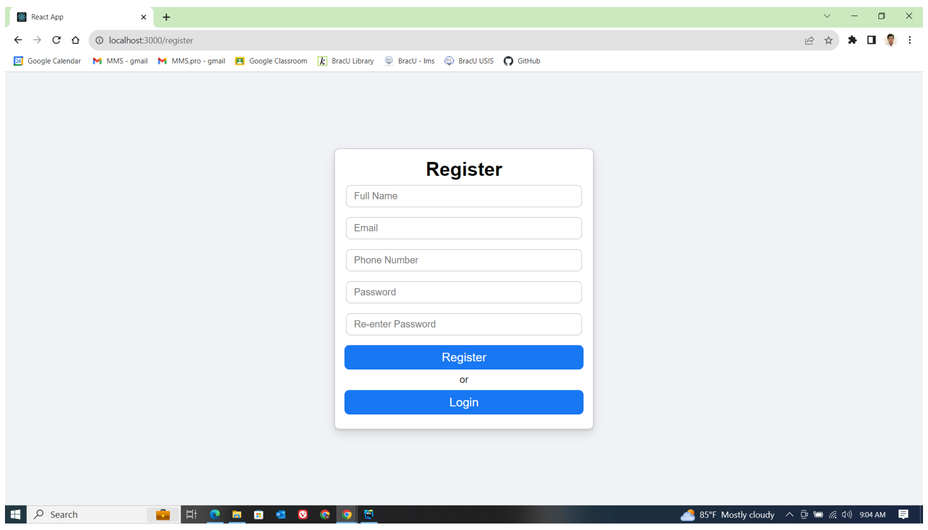
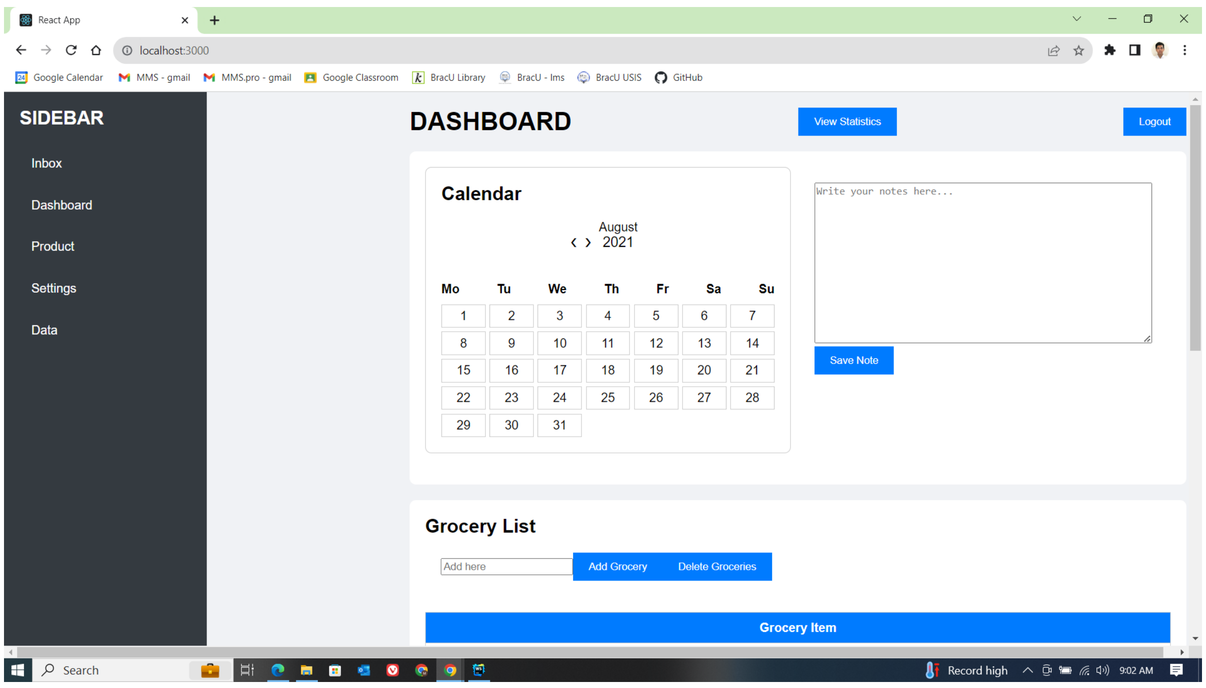


Fig2: The Registration interface of the website

Dashboard

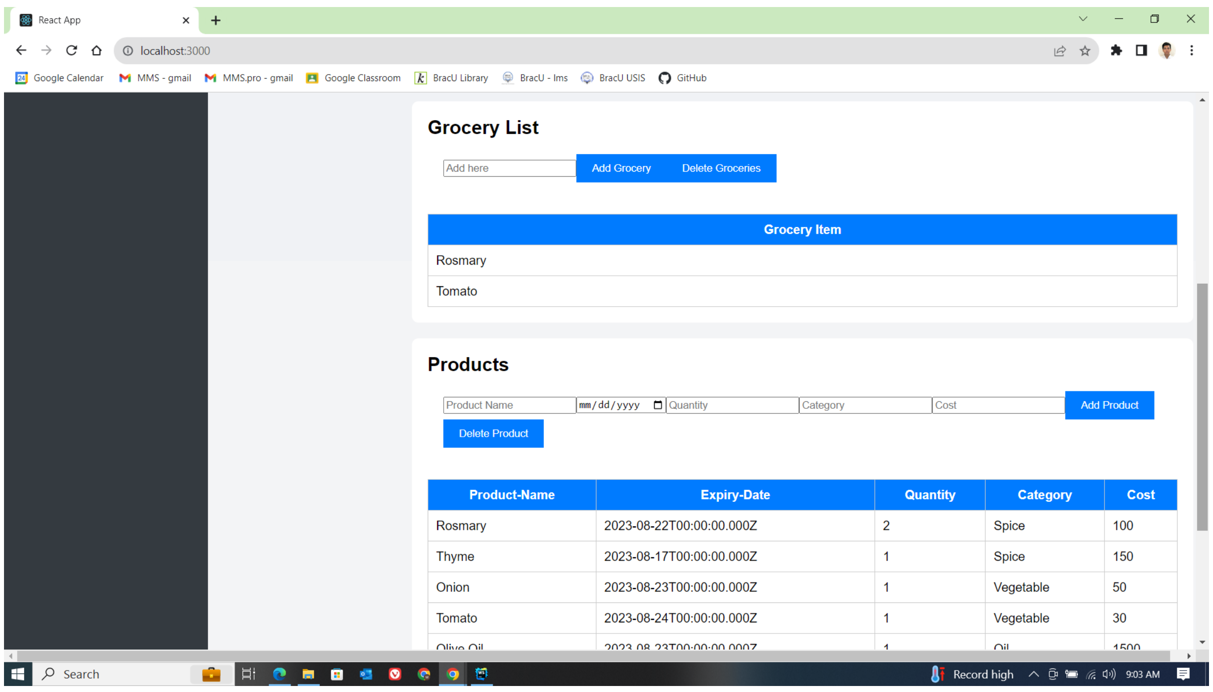
- ***<u>Calendar:</u>*** Once the user is logged in, they see Figure 3 initially. After they scroll down, they get to see figures 4 and 5. The dashboard provides a holistic view of the website. Here we have a calendar to see which day it is. We can add a note here for the user to remind themselves of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



Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

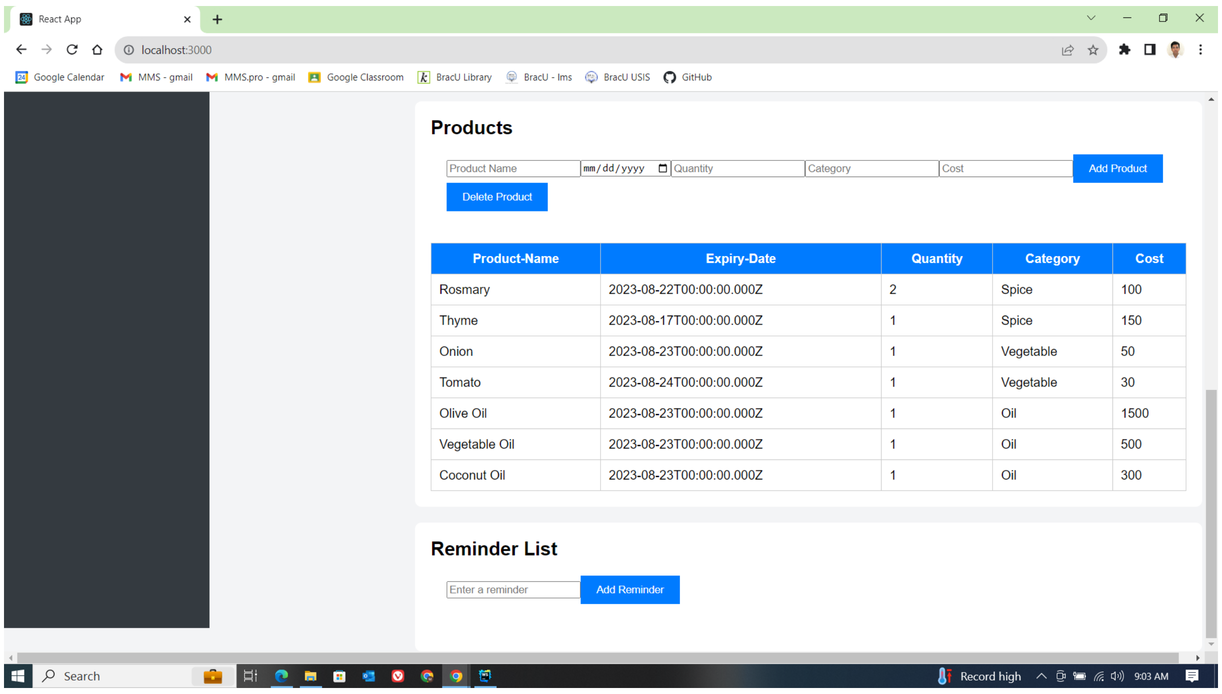
Fig3: The dashboard-1



Grocery Item
Rosmary
Tomato

Product-Name	Expiry-Date	Quantity	Category	Cost
Rosmary	2023-08-22T00:00:00.000Z	2	Spice	100
Thyme	2023-08-17T00:00:00.000Z	1	Spice	150
Onion	2023-08-23T00:00:00.000Z	1	Vegetable	50
Tomato	2023-08-24T00:00:00.000Z	1	Vegetable	30
Olive Oil	2023-08-23T00:00:00.000Z	1	Oil	1500

Fig4: The dashboard-2



Products

Product Name | mm/dd/yyyy | Quantity | Category | Cost | Add Product

Delete Product

Product-Name	Expiry-Date	Quantity	Category	Cost
Rosmary	2023-08-22T00:00:00.000Z	2	Spice	100
Thyme	2023-08-17T00:00:00.000Z	1	Spice	150
Onion	2023-08-23T00:00:00.000Z	1	Vegetable	50
Tomato	2023-08-24T00:00:00.000Z	1	Vegetable	30
Olive Oil	2023-08-23T00:00:00.000Z	1	Oil	1500
Vegetable Oil	2023-08-23T00:00:00.000Z	1	Oil	500
Coconut Oil	2023-08-23T00:00:00.000Z	1	Oil	300

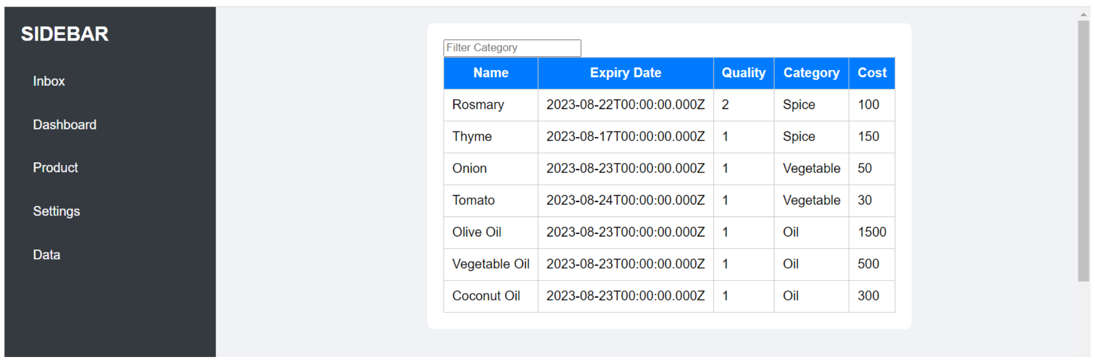
Reminder List

Enter a reminder | Add Reminder

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.

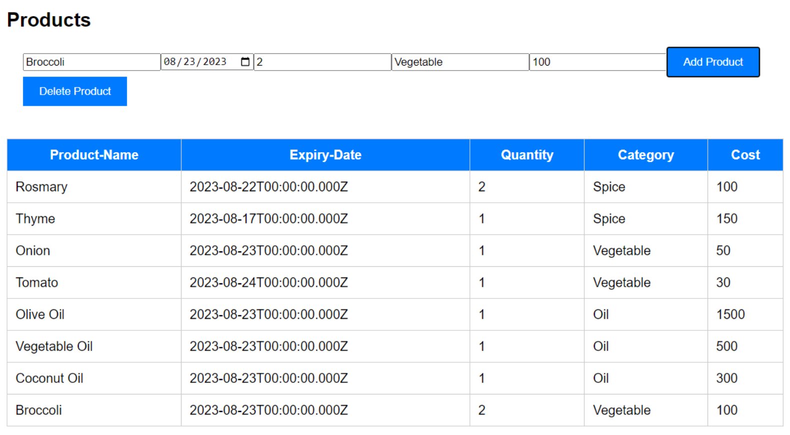


Filter Category

Name	Expiry Date	Quality	Category	Cost
Rosmary	2023-08-22T00:00:00.000Z	2	Spice	100
Thyme	2023-08-17T00:00:00.000Z	1	Spice	150
Onion	2023-08-23T00:00:00.000Z	1	Vegetable	50
Tomato	2023-08-24T00:00:00.000Z	1	Vegetable	30
Olive Oil	2023-08-23T00:00:00.000Z	1	Oil	1500
Vegetable Oil	2023-08-23T00:00:00.000Z	1	Oil	500
Coconut Oil	2023-08-23T00:00:00.000Z	1	Oil	300

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.



Products

Broccoli | 08/23/2023 | 2 | Vegetable | 100 | Add Product

Delete Product

Product-Name	Expiry-Date	Quantity	Category	Cost
Rosmary	2023-08-22T00:00:00.000Z	2	Spice	100
Thyme	2023-08-17T00:00:00.000Z	1	Spice	150
Onion	2023-08-23T00:00:00.000Z	1	Vegetable	50
Tomato	2023-08-24T00:00:00.000Z	1	Vegetable	30
Olive Oil	2023-08-23T00:00:00.000Z	1	Oil	1500
Vegetable Oil	2023-08-23T00:00:00.000Z	1	Oil	500
Coconut Oil	2023-08-23T00:00:00.000Z	1	Oil	300
Broccoli	2023-08-23T00:00:00.000Z	2	Vegetable	100

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.

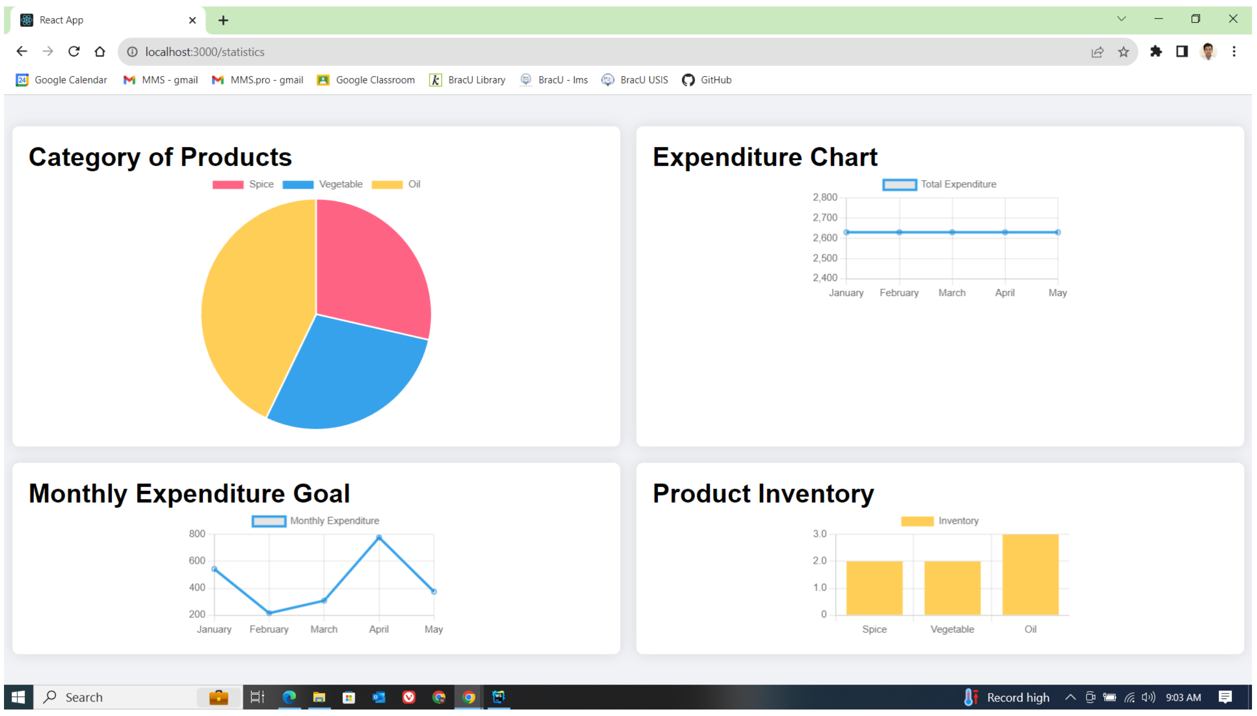


Fig8: The Statistics Page

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.

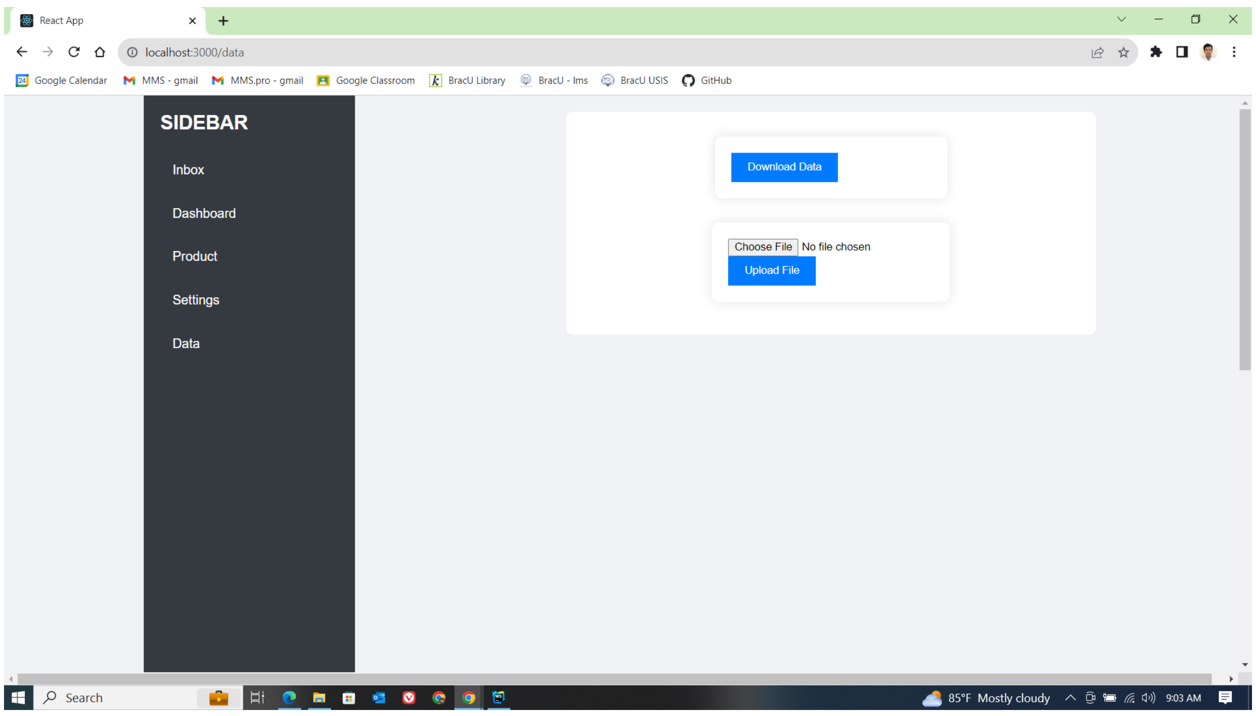
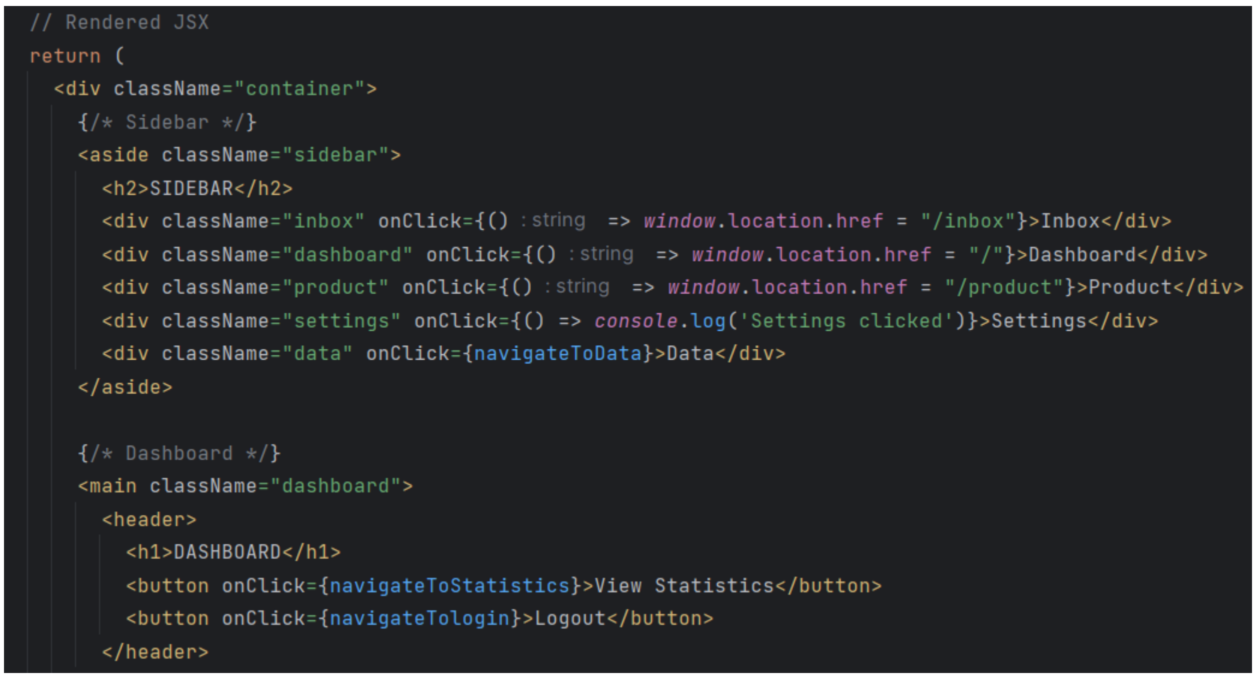


Fig9: The Data Manipulation Page

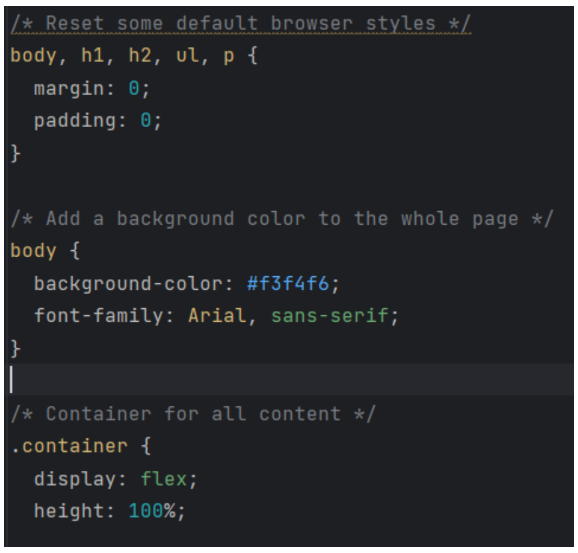
Front End Development

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:



```
// Rendered JSX
return (
  <div className="container">
    {/* Sidebar */}
    <aside className="sidebar">
      <h2>SIDEBAR</h2>
      <div className="inbox" onClick={() :string  => window.location.href = "/inbox"}>Inbox</div>
      <div className="dashboard" onClick={() :string  => window.location.href = "/"}>Dashboard</div>
      <div className="product" onClick={() :string  => window.location.href = "/product"}>Product</div>
      <div className="settings" onClick={() => console.log('Settings clicked')}>Settings</div>
      <div className="data" onClick={navigateToData}>Data</div>
    </aside>

    {/* Dashboard */}
    <main className="dashboard">
      <header>
        <h1>DASHBOARD</h1>
        <button onClick={navigateToStatistics}>View Statistics</button>
        <button onClick={navigateTologin}>Logout</button>
      </header>
```

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

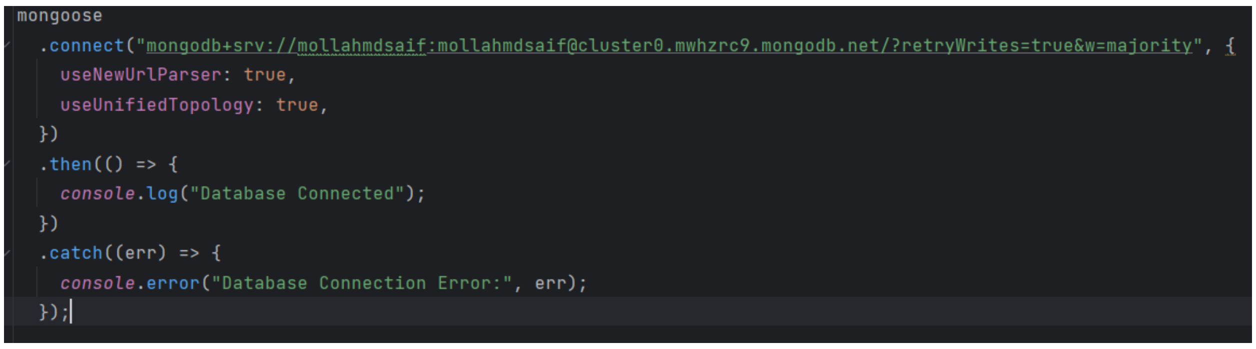


```
/* Reset some default browser styles */
body, h1, h2, ul, p {
  margin: 0;
  padding: 0;
}

/* Add a background color to the whole page */
body {
  background-color: #f3f4f6;
  font-family: Arial, sans-serif;
}

/* Container for all content */
.container {
  display: flex;
  height: 100%;
```

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

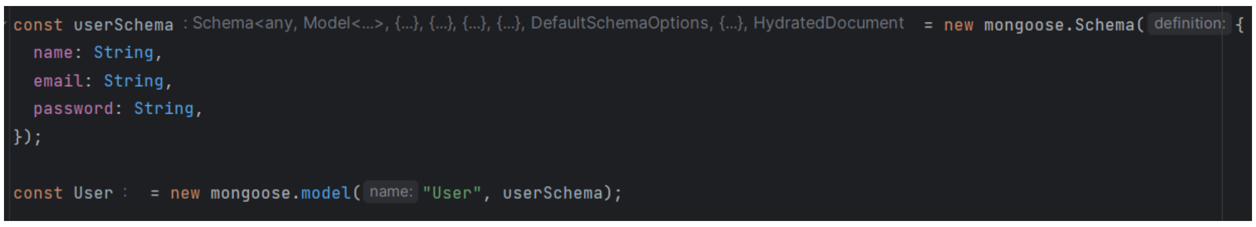
Javascript is used for our back-end programming language. For keeping creating Routing, Middleware, and API easily we used Express Framework. We have used MongoDB for our database. MongoDB allowed us to host the database in the cloud and connect with the Express server very conveniently. The code snippet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



```
mongoose
  .connect("mongodb+srv://mollahmdsaif:mollahmdsaif@cluster0.mwhzrc9.mongodb.net/?retryWrites=true&w=majority", {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  })
  .then(() => {
    console.log("Database Connected");
  })
  .catch((err) => {
    console.error("Database Connection Error:", err);
  });
```

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:



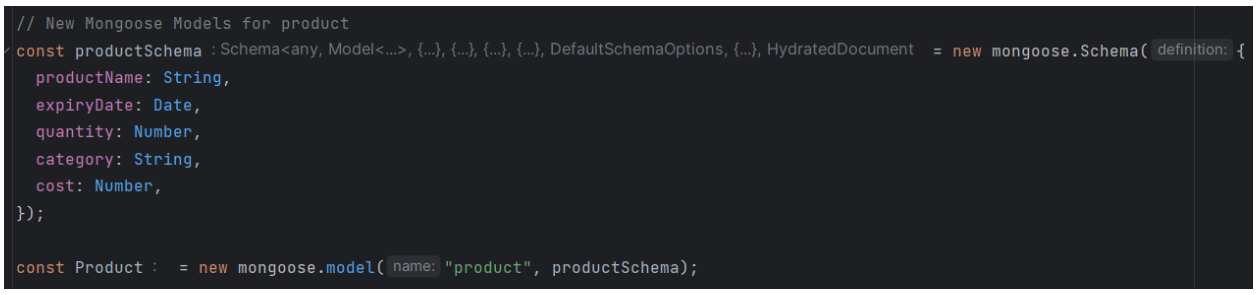
```
const userSchema :Schema<any, Model<...>, {...}, {...}, {...}, {...}, DefaultSchemaOptions, {...}, HydratedDocument  = new mongoose.Schema( definition: {
  name: String,
  email: String,
  password: String,
});

const User :  = new mongoose.model( name: "User", userSchema);
```

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:



```
// New Mongoose Models for product
const productSchema :Schema<any, Model<...>, {...}, {...}, {...}, {...}, DefaultSchemaOptions, {...}, HydratedDocument  = new mongoose.Schema( definition: {
  productName: String,
  expiryDate: Date,
  quantity: Number,
  category: String,
  cost: Number,
});

const Product :  = new mongoose.model( name: "product", productSchema);
```

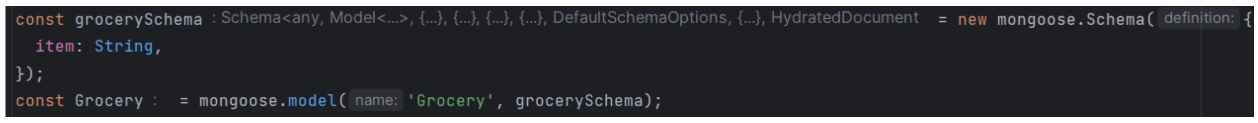
Using the product object we use three API functions are

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:



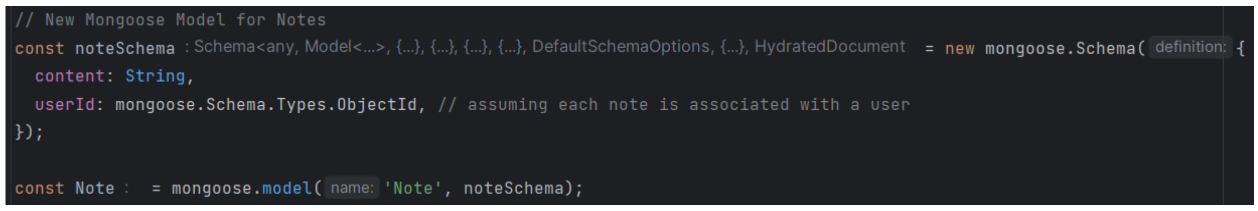
```
const grocerySchema : Schema<any, Model<...>, {...}, {...}, {...}, {...}, DefaultSchemaOptions, {...}, HydratedDocument  = new mongoose.Schema( definition: {
  item: String,
});
const Grocery :  = mongoose.model( name: 'Grocery', grocerySchema);
```

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:



```
// New Mongoose Model for Notes
const noteSchema : Schema<any, Model<...>, {...}, {...}, {...}, {...}, DefaultSchemaOptions, {...}, HydratedDocument  = new mongoose.Schema( definition: {
  content: String,
  userId: mongoose.Schema.Types.ObjectId, // assuming each note is associated with a user
});

const Note :  = mongoose.model( name: 'Note', noteSchema);
```

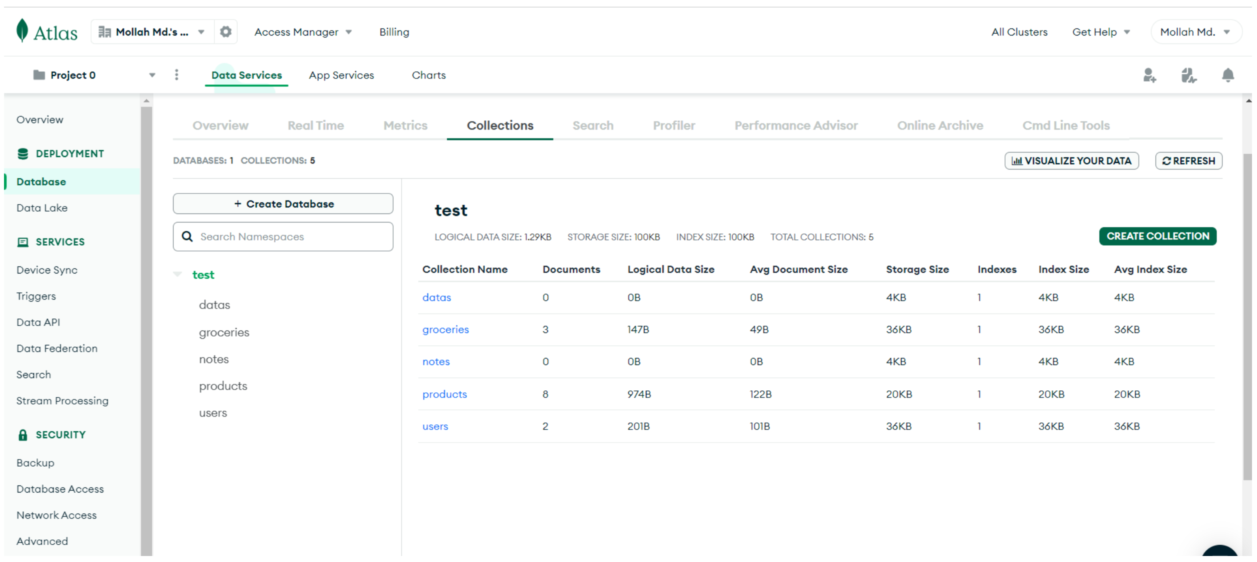
Using the note object we use three API functions are:

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

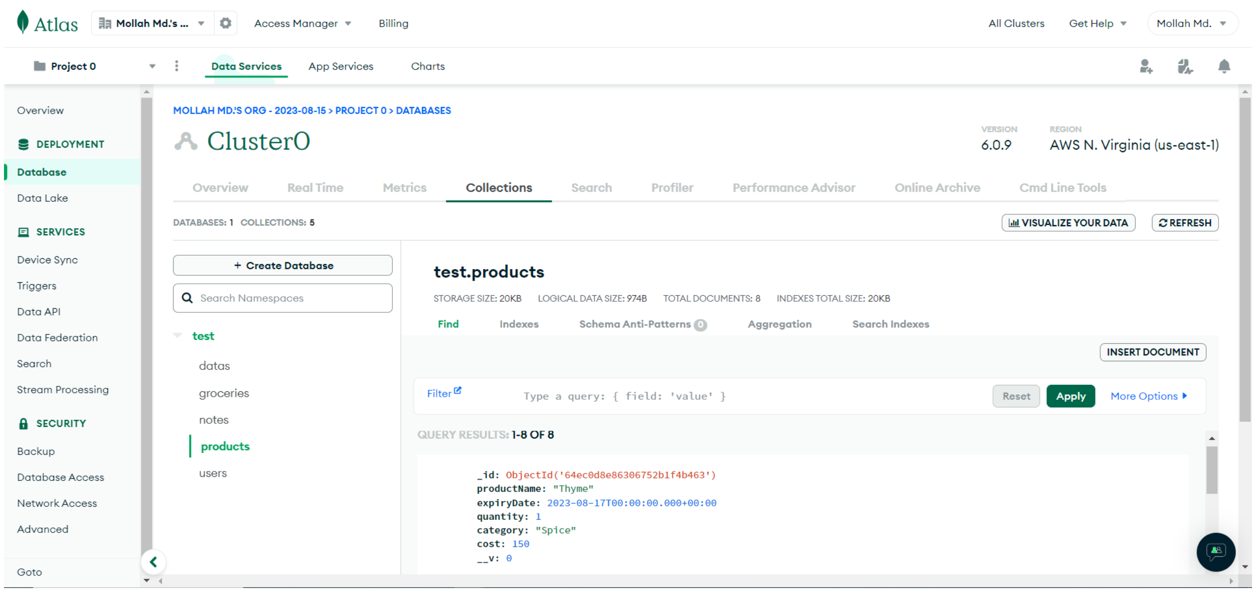


DATABASES: 1 COLLECTIONS: 5

test

LOGICAL DATA SIZE: 1.29KB STORAGE SIZE: 100KB INDEX SIZE: 100KB TOTAL COLLECTIONS: 5

Collection Name	Documents	Logical Data Size	Avg Document Size	Storage Size	Indexes	Index Size	Avg Index Size
datas	0	0B	0B	4KB	1	4KB	4KB
groceries	3	147B	49B	36KB	1	36KB	36KB
notes	0	0B	0B	4KB	1	4KB	4KB
products	8	974B	122B	20KB	1	20KB	20KB
users	2	201B	101B	36KB	1	36KB	36KB



MOLLAH MD.'S ORG - 2023-08-15 > PROJECT 0 > DATABASES

Cluster0

VERSION 6.0.9 REGION AWS N. Virginia (us-east-1)

DATABASES: 1 COLLECTIONS: 5

test.products

STORAGE SIZE: 20KB LOGICAL DATA SIZE: 974B TOTAL DOCUMENTS: 8 INDEXES TOTAL SIZE: 20KB

Type a query: { field: 'value' }

QUERY RESULTS: 1-8 OF 8

```
_id: ObjectId('64ec0d8e86306752b1f4b463')
productName: "Thyme"
expiryDate: 2023-08-17T00:00:00.000+00:00
quantity: 1
category: "Spice"
cost: 150
__v: 0
```

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	Feature Engineering, Sprint Planning, Software Requirement Specification
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