# NUTRITION OVERVIEW

### ABSTRACT

This project focuses on creating a web application to provide effective way to review and analysis the nutrition based on the composition intake. Based on the massive nutrition information, it would be interesting to understand what are the important factors that make a nutrition suggestion more successful than others. So, we would like to analyze what kind of nutrition are required for the individuals and check the food diet they required to maintain during any circumstances like weight loss, weight gain, health issues time food to eat.

In this project, we take nutrition compositions like food group, calories, fat, protein, carbohydrates, saturated fats, fiber, cholesterol as response variable and focus on operating predictions by analyzing the rest of variables in the 1.8K data records. The results can help people can understand the foods they want to consume by considering the criteria (like protein, fats etc.,)

## TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Title** | **Page No.** |
|  | **Abstract** | **1** |
| **1**  **1.1**  **1.2** | **Introduction**  **Objectives**  **Background and Literature Survey** | **3**  **3**  **3** |
| **2**  **2.1**  **2.2** | **Describing movie overview**  **Proposed System**  **Working Methodology** | **4**  **4**  **5** |
| **3**  **3.1** | **Database Analysis**  **Dataset used and attributes** | **6**  **6** |
| **4** | **Results and Discussion** | **7** |
| **5** | **Conclusion & Future Works** | **12** |
| **6** | **Appendix** | **13** |
| **7** | **References** | **21** |

**CHAPTER 1**

### INTRODUCTION

Nutritionists use ideas from molecular biology, biochemistry, and genetics to understand how nutrients affect the human body. Nutrition also focuses on how people can use dietary choices to reduce the risk of disease, what happens if a person has too much or too little of a nutrient, and how allergies work. Nutrients provide nourishment. Proteins, carbohydrates, fat, vitamins, minerals, fiber, and water are all nutrients. If people do not have the right balance of nutrients in their diet, their risk of developing certain health conditions increases.

#### Objectives

The following are the objectives of this project:

* creating an application to provide effective assistance to review foods
* Simple UI that the user can able to easily understandable.
* Search filters to make up the process of reviewing quicker.
* Deletion and updating if any wrong data can do easily.

#### Background and Literature Survey

Perspectives and practices related to food and nutrition have greatly changed from the ancient era to today. In the ancient world, location and economic status had a profound effect on what people ate. Also, societies often were based on crop cultivation and livestock rearing, which influenced how people ate, worked, and lived. During the Medieval Era, people became more exposed to food from other parts of the world because of the growing ability to ship goods among other factors.

Some nutrients also act as antioxidants. These may be vitamins, minerals, proteins, or other types of molecules. They help the body remove toxic substances known as free radicals, or reactive oxygen species. If too many of these substances remain in the body, cell damage and disease can result.

### Describing the Movie over-view

In this describes the proposed system, working methodology and software details.

#### 2.1 Proposed System

The following diagram shows the system architecture of this project.

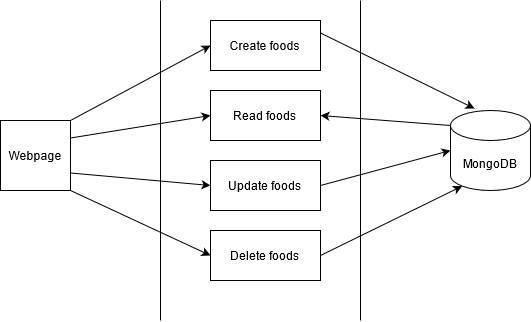
****

Figure 1. System Architecture Block Diagram

#### 2.2 Working Methodology

The project is completely based on software with two sections, front-end and back-end. The front-end consists of an HTML(HyperText Markup Language) file and a CSS(Cascading Style Sheets) file.

This is build using the python, flask framework (is a popular Python web framework, meaning it is a third-party Python library used for developing web applications), mongo DB database sever and compass (GUI).

The flask helps us to create dynamic webpages and connects to the port 5000. For the local hosting <http://localhost:5000>. We need run the mongo server to connect with DB and do the CRUD operations. Due the web applications the crud operations are made as simple to everyone.

Logins of Admins and Users and new registrations are also possible and given authentication access to users and admins. The admin can do all CRUD operations to data for the users to use to make their nutrition diet and add their requires foods and make their dietary palns.

The creation of the data and fetching the data as by desired using the constraints we required and even update and delete the data even after applying the filters. Thus, we can make the review based many overviews of the data and able to come to the conclusion.

**CHAPTER 3**

### DATABASE ANALYSIS

#### 3.1 Datasets and attributes

Nutrition Analysis dataset is used and the attributes like name of the food, food\_group of the food, calories in the food, fat in the food, protein in the food, carbohydrates in the food, saturated fats of the food, fiber in the food.

This dataset taken form Wikipedia nutrition and USDA nutrition website taken reference from the many sites and collected data and filtered it to our requirement.

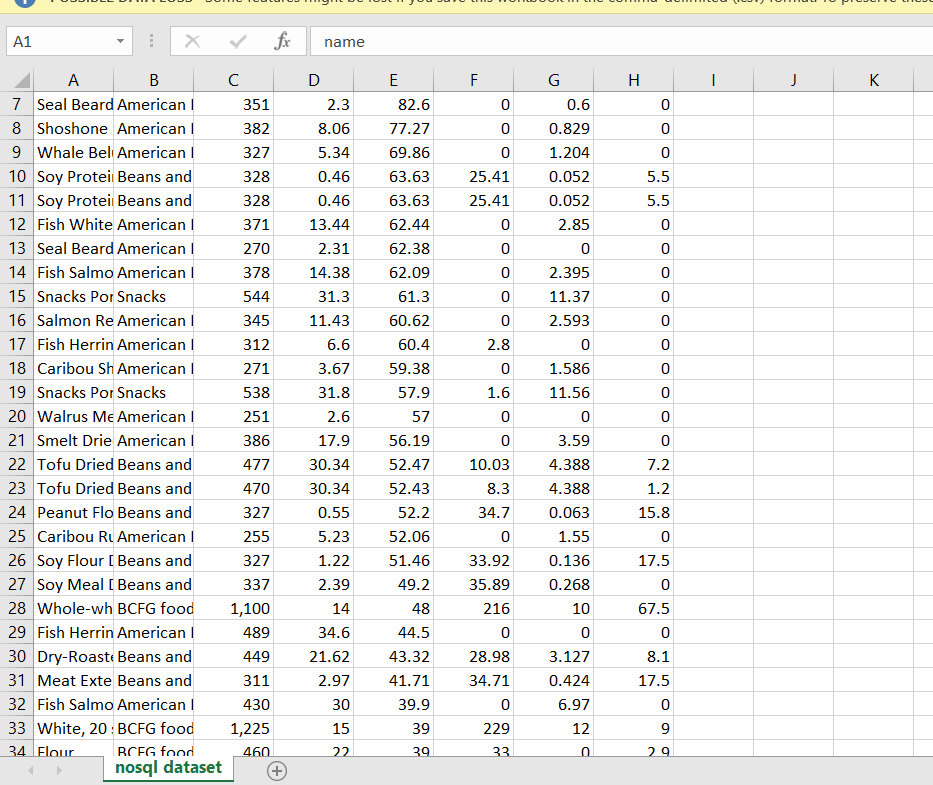


Fig: 1 Dataset

**CHAPTER 4**

### RESULTS AND DISCUSSIONS

The application consists of a two-web pages allowing the user to add the film details and another to fetch & read the desired data or whole data.

The web page (fig: 3) show the whole details and even able allow to filter the movie data based on name, year, runtime, genre, rating, meta score, timeline, votes, gross. Thus make wide variety of decisions. The next index (fig: 2) is to simply add the movie data and that stores in the DB.

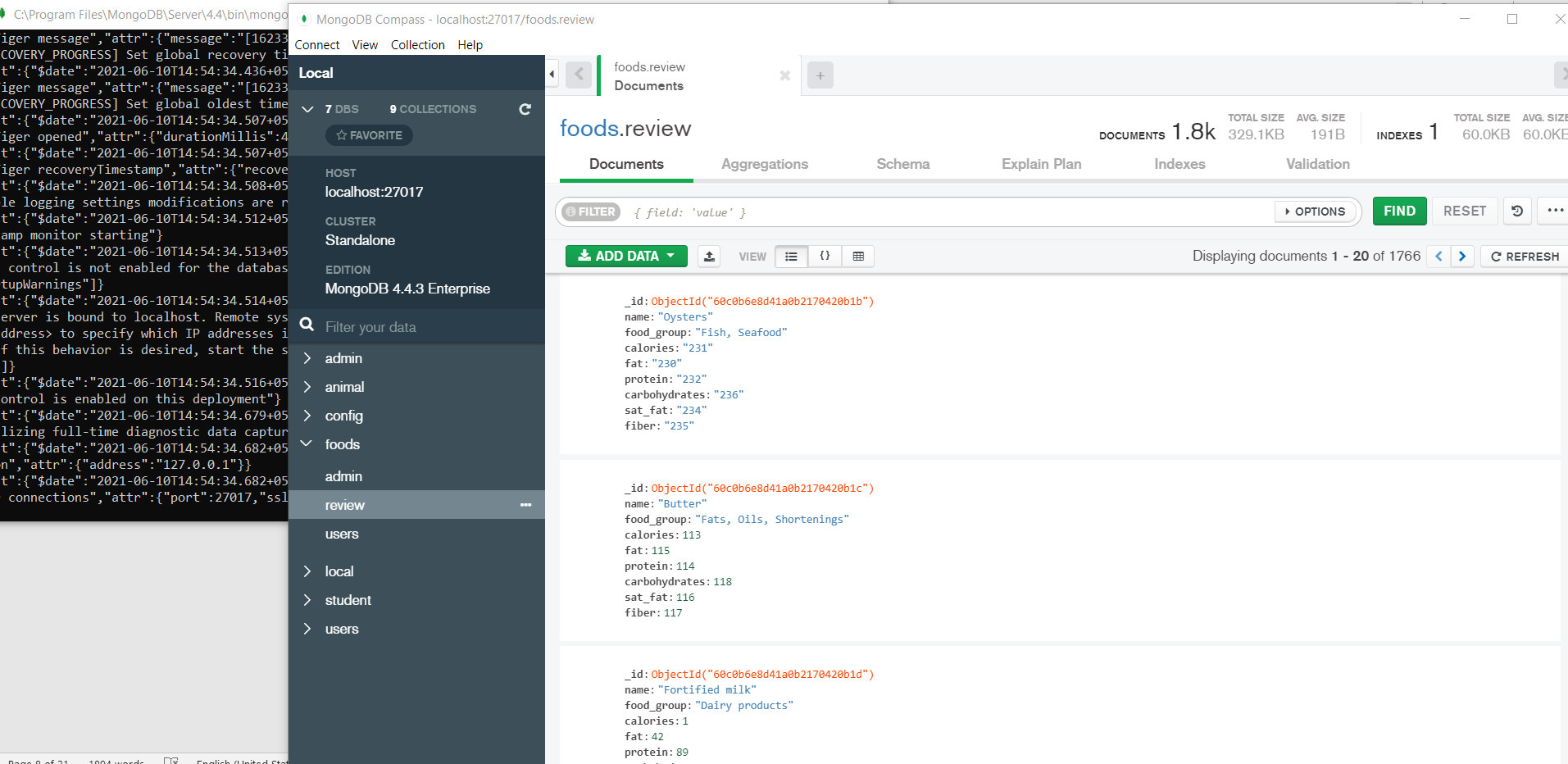


Fig: 1 MongoDB connection and view of DB

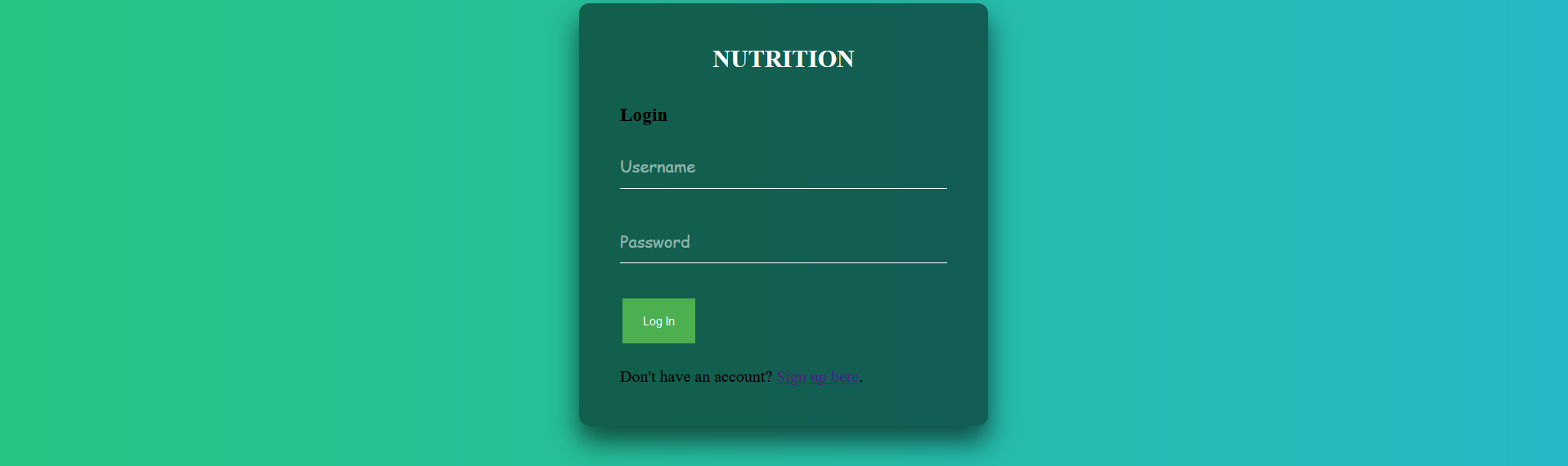
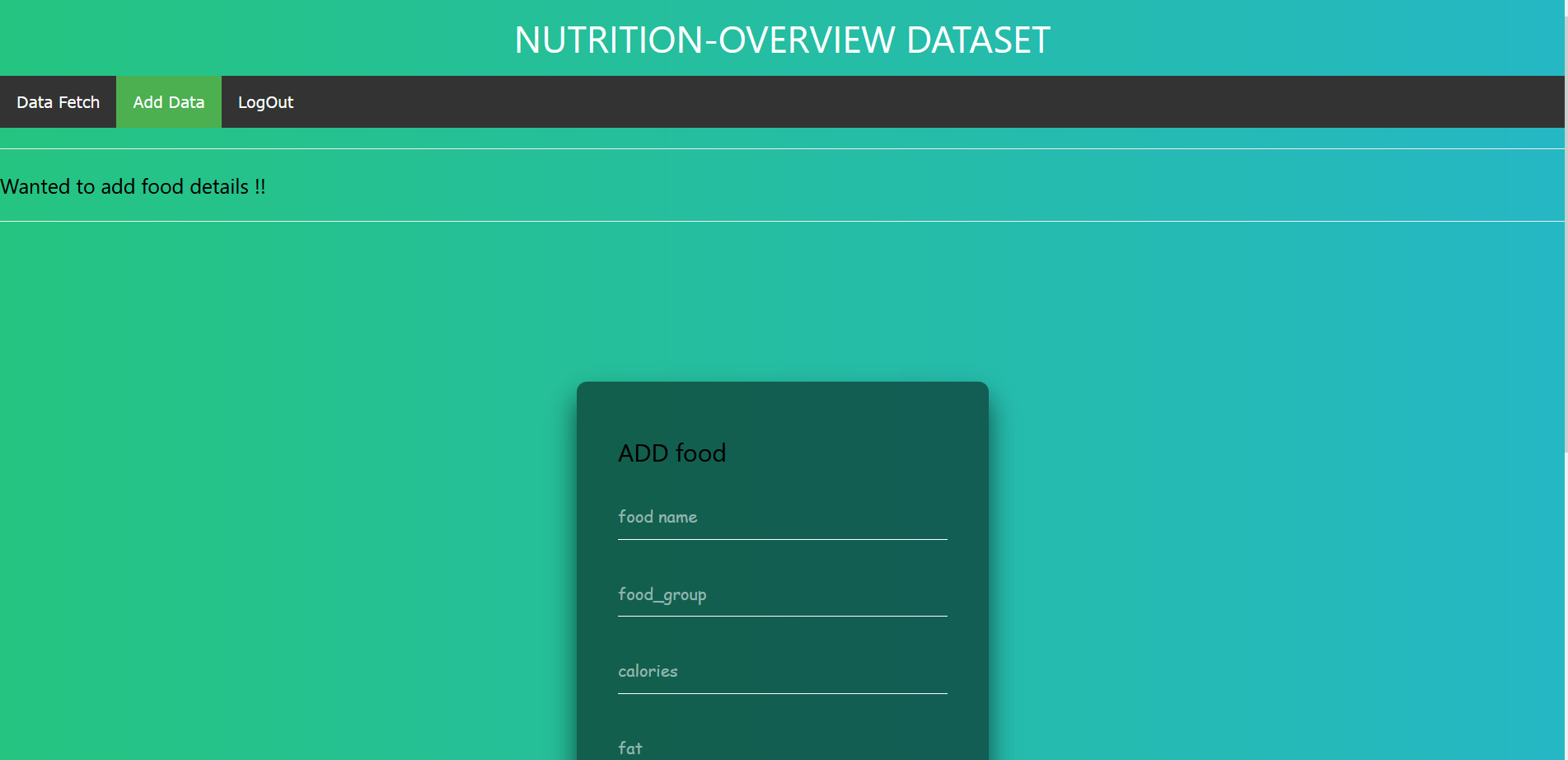


Fig : Login Page



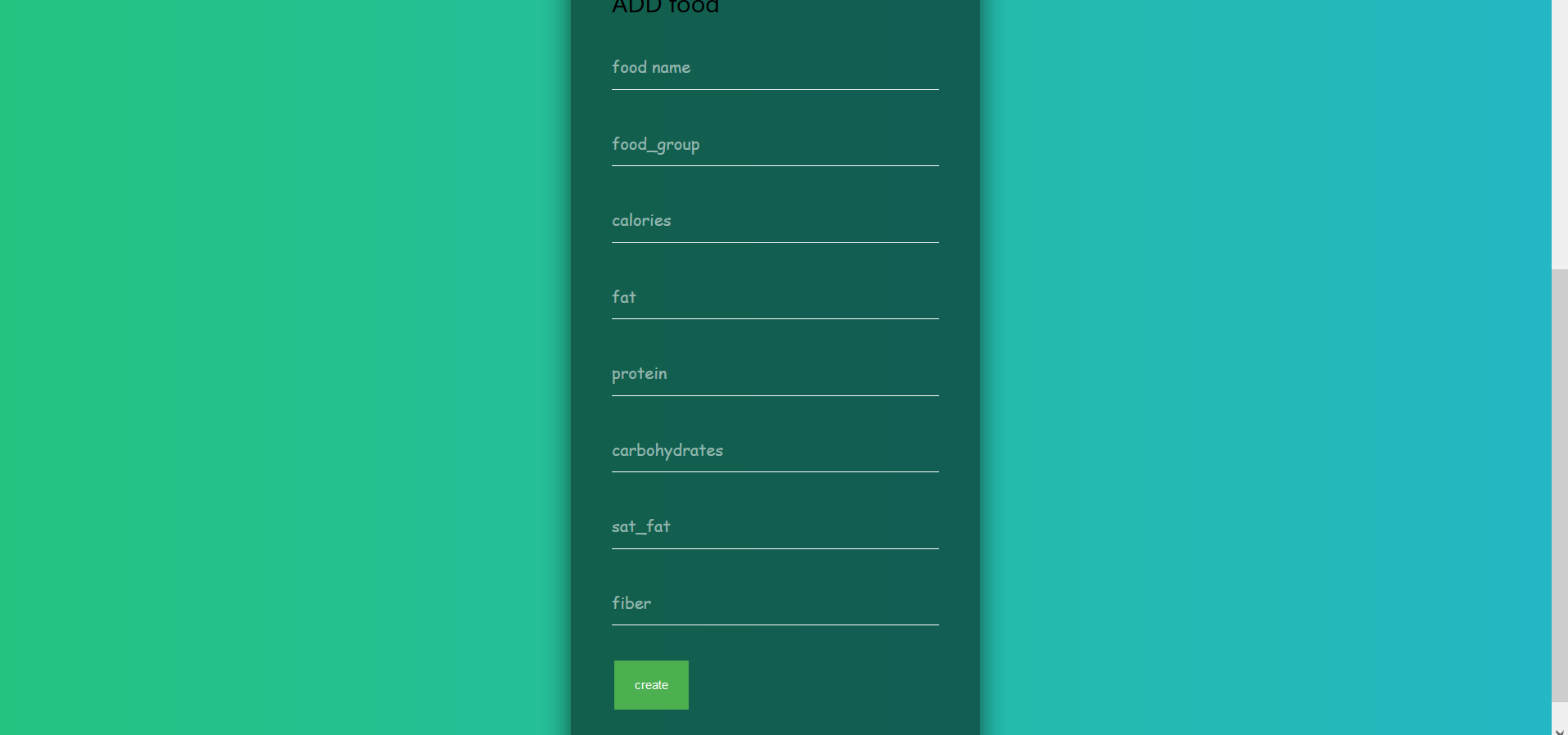
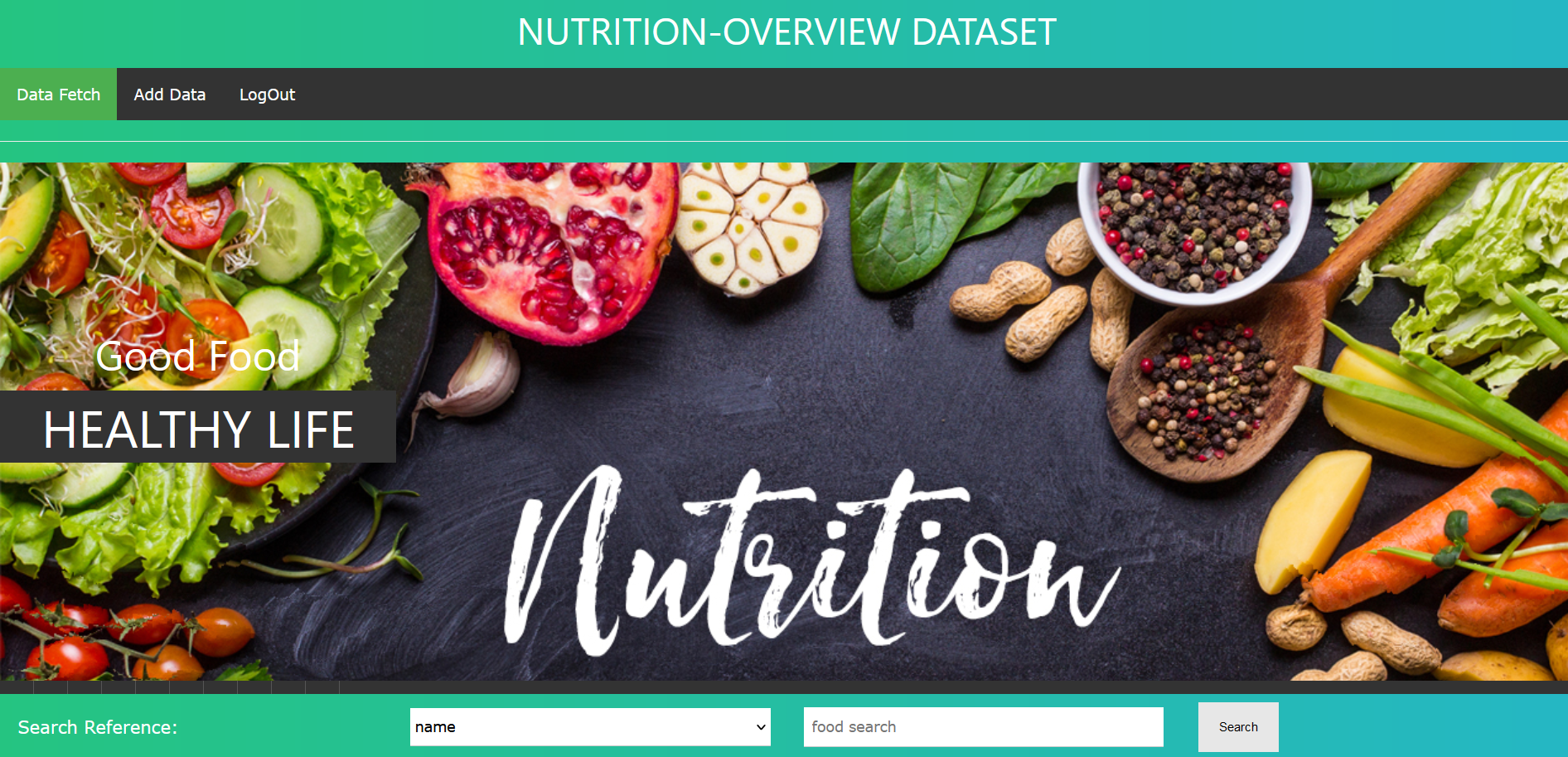


Fig: 2 Index page



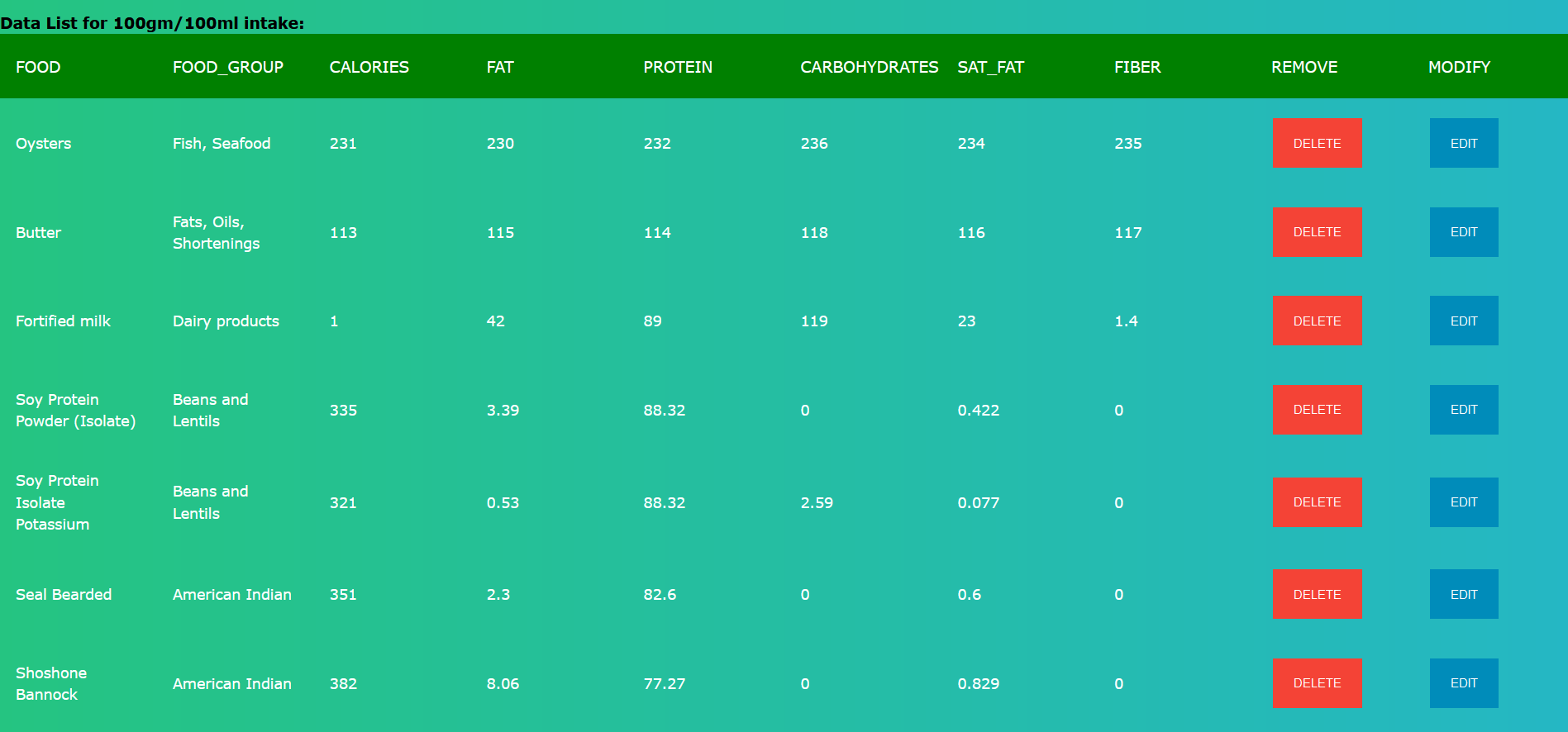


Fig: 3 fetch the data & filter

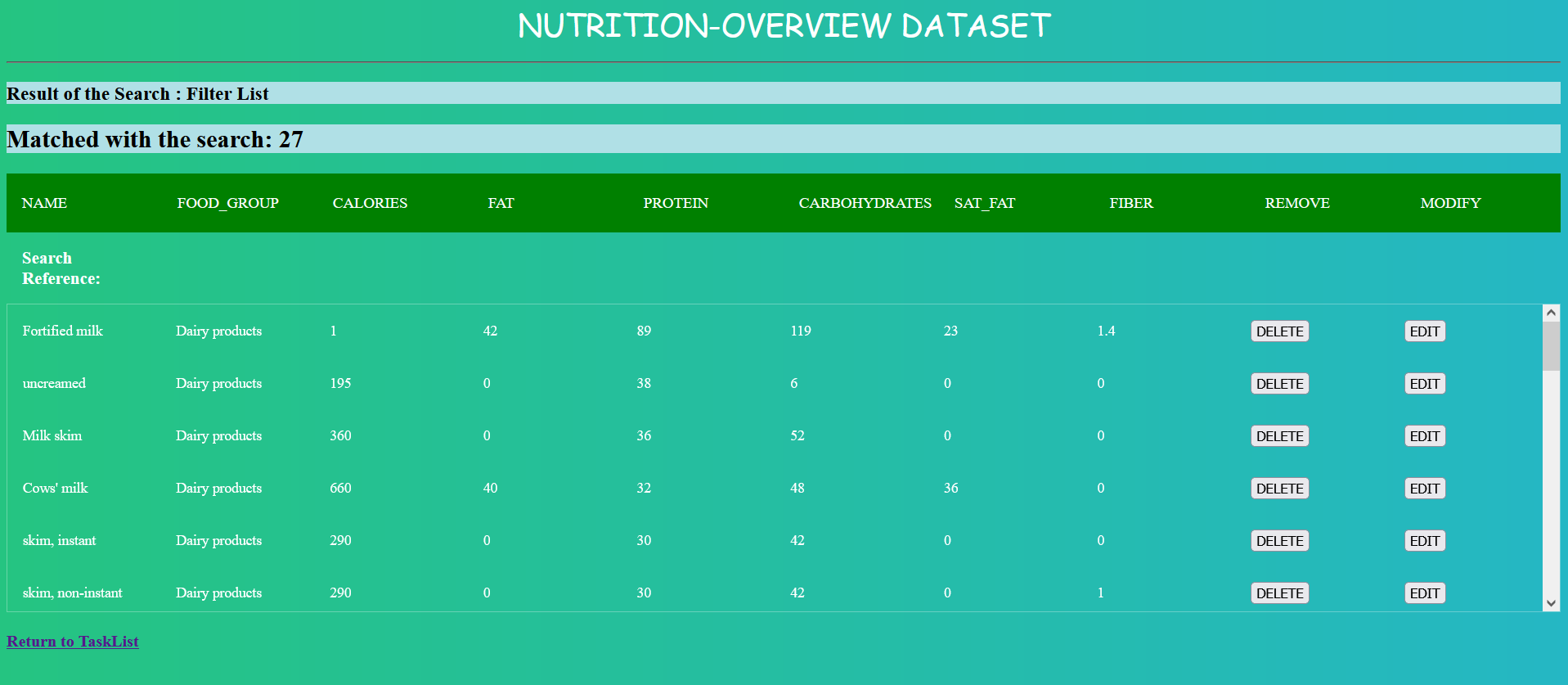


Fig: 4 filter by Dairy products

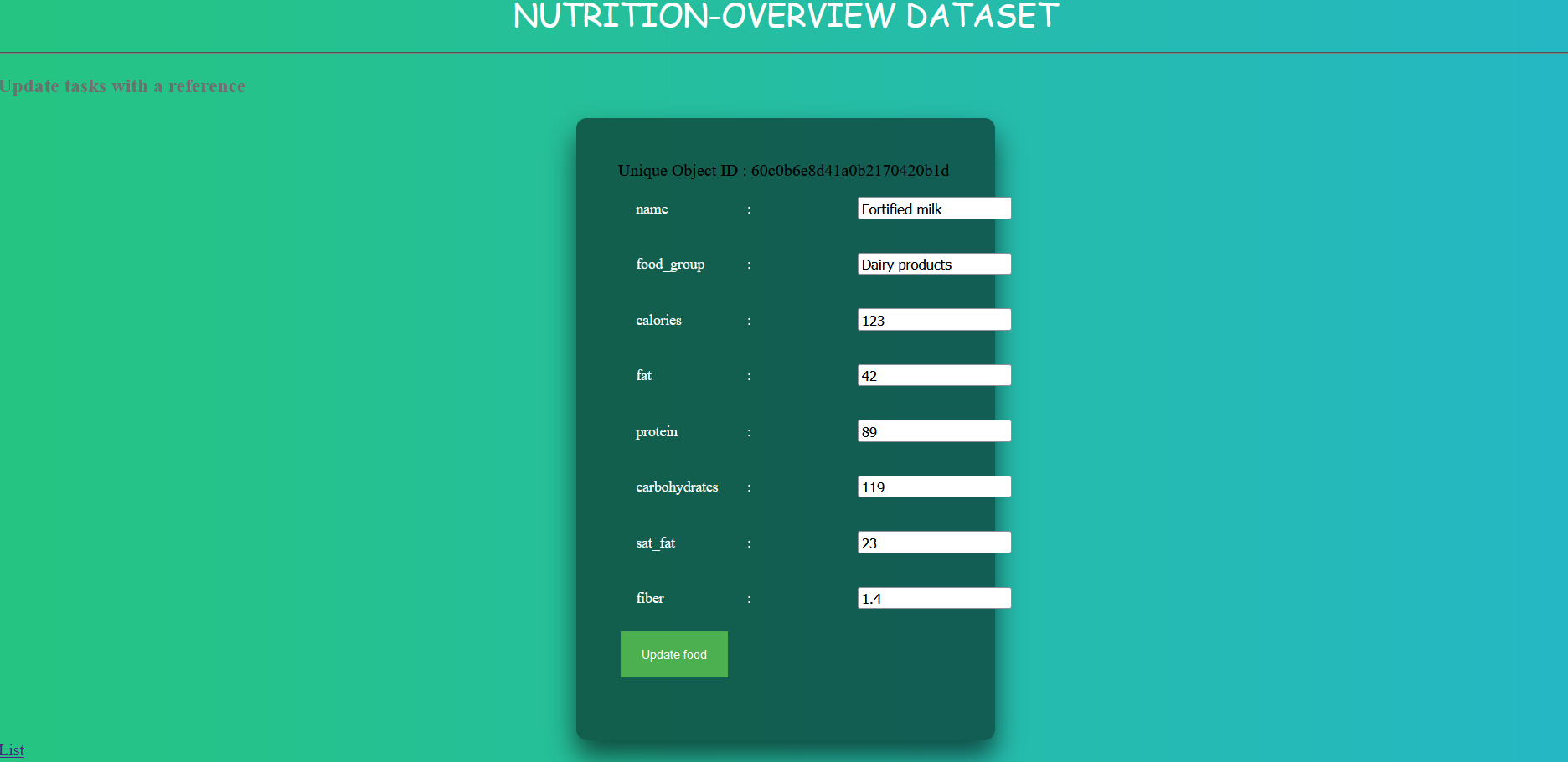
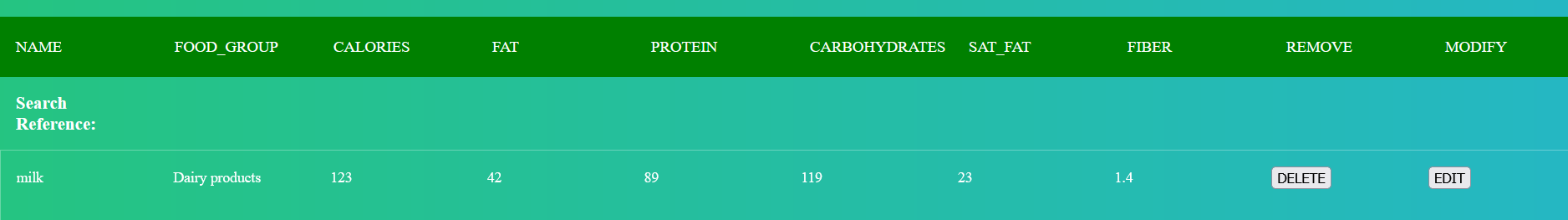


Fig: 5 Updating of the data



After update

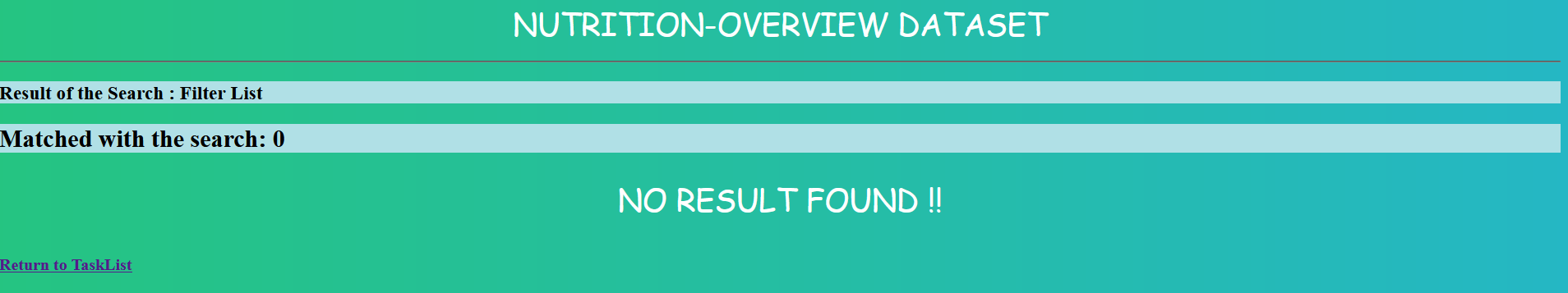


Fig: 6 After deletion

**CHAPTER 5**

### CONCLUSION AND FUTURE WORK

This web application builds a path to decide the right nutrition diet making decision based on differentiating on various on parameters and able to make the good diet plan by makers and the able to make decision by the diet lover to the food based on the weight gain and weight loss and for nutrition.

Lot can be done in this area. This project can be extended

* To add more parameters so that decisions can be made by considering huge parameters circumstances.
* to automate the entry of the movie data, when ever the new food is released and store the review using api’s.
* to add to cloud, such the even the data entry increase and there would be no issues of the storage.

**CHAPTER 6**

### APPENDIX

Front-end (HTML) - index.html

<html>

    <head>

        <title>{{t}}</title>

        <!-- href="/static/assets/style.css"-->

        <link rel="stylesheet" type="text/css"  href="{{ url\_for('static',filename='assets/style.css')}}" >

        <link rel="stylesheet" type="text/css"  href="{{ url\_for('static',filename='assets/emoji.css')}}" >

        <script src="{{ url\_for('static',filename='assets/twemoji.min.js')}}"></script>

        <script src="{{ url\_for('static',filename='assets/emoji.js')}}"></script>

    </head>

<body>

    <h1>{{ h }}</h1>

    <ul>

        <li><a href="/list" class="{{ a1 }}">Data Fetch</a></li>

        <li><a href="/" class="{{ a2 }}">Add Data</a></li>

        <!--<li><a href="/completed" class="{{ a3 }}">Completed</a></li>-->

    </ul>

    <hr>

    {% if todos[0] %}

    <div span="right">

    <form action="/search"  method="GET" >

        <table class="none" id="close">

        <tr>

        <td></td><td></td>

        <td><big><b>Search Reference:</b></big></td>

        <td><select name="refer" required>

            <option value="name">name</option>

            <option value="year">year</option>

            <option value="runtime">runtime</option>

            <option value="genre">genre</option>

            <option value="rating">rating</option>

            <option value="metascore">metascore</option>

            <option value="timeline">timeline</option>

            <option value="votes">votes</option>

            <option value="gross">gross</option>

        </select></td>

        <td><input type="text" name="key" placeholder="movie search" size="15" /></td>

        <td><button type="submit">Search</button></td>

        </tr>

        </table>

    </form>

    </div>

    <b><big>Data List :</big></b>

    <table>

        <tr id="row">

            <th class="name">name</th>

            <th class="year">year</th>

            <th class="runtime">runtime</th>

            <th class="genre">genre</th>

            <th class="rating">ratinge</th>

            <th class="metascore">metascore</th>

            <th class="timeline">timeline</th>

            <th class="votes">votes</th>

            <th class="gross">gross</th>

        <th class="func1">Remove</th>

        <th class="func2">Modify</th>

        </tr>

    {% for todo in todos %}

        <tr class="datas">

            <td class="name">{{ todo["name"] }}</td>

            <td class="year">{{ todo["year"] }}</td>

            <td class="date">{{ todo["runtime"] }}</td>

            <td class="genre">{{ todo["genre"] }}</td>

            <td class="rating">{{ todo["rating"] }}</td>

            <td class="metascore">{{ todo["metascore"] }}</td>

            <td class="timeline">{{ todo["timeline"] }}</td>

            <td class="votes">{{ todo["votes"] }}</td>

            <td class="gross">{{ todo["gross"] }}</td>

            <td class="func1"><a href="./remove?\_id={{ todo['\_id'] }}"><button type="submit">DELETE</button></a></td>

            <td class="func1"><a href="./update?\_id={{ todo['\_id'] }}"><button type="submit">EDIT</button></a></td>

        </tr>

    {% endfor %}

    </table>

    {% else %}

        <h4>Wanted to add movie details !!</h4>

    {% endif %}

    <hr/>

    <form action="/action" method="POST">

        <table class="none">

            <tr>

                <td><b><big><label>Add a Movie : </label></big></b></td>

            </tr>

            <tr>

            <td><input type="text" name="name" placeholder="Movie name" /></td>

            <td><input type="text" name="year" placeholder="year" /></td>

            <td><input type="text" name="runtime" placeholder="runtime" /></td>

            <td><input type="text" name="genre" placeholder="genre" /></td>

            <td><input type="text" name="rating" placeholder="rating" /></td>

            <td><input type="text" name="metascore" placeholder="metascore" /></td>

            <td><input type="text" name="timeline" placeholder="timeline" /></td>

            <td><input type="text" name="votes" placeholder="votes" /></td>

            <td><input type="text" name="gross" placeholder="gross" /></td>

            <td><button type="submit"> Create </button></td>

            </tr>

        </form>

        </table>

</body>

</html>

Front-end : Search.html

<html>

    <head>

        <title>{{t}}</title>

        <link rel="stylesheet" type="text/css"  href="{{ url\_for('static',filename='assets/style.css')}}" >

    </head>

<body>

    <h1>{{h}}</h1>

    <hr>

    {% if todos[0] %}

    <h3>Result of the Search : Filter List</h3>

    <table>

        <tr id="row">

        <th class="name">name</th>

        <th class="year">year</th>

            <th class="runtime">runtime</th>

            <th class="genre">genre</th>

            <th class="rating">ratinge</th>

            <th class="metascore">metascore</th>

            <th class="timeline">timeline</th>

            <th class="votes">votes</th>

            <th class="gross">gross</th>

        <td class="func">Delete</td>

        <td class="func">Modify</td>

        </tr>

        {% for todo in todos %}

        <tr class="datas">

            <td class="name">{{ todo["name"] }}</td>

            <td class="year">{{ todo["year"] }}</td>

            <td class="date">{{ todo["runtime"] }}</td>

            <td class="genre">{{ todo["genre"] }}</td>

            <td class="rating">{{ todo["rating"] }}</td>

            <td class="metascore">{{ todo["metascore"] }}</td>

            <td class="timeline">{{ todo["timeline"] }}</td>

            <td class="votes">{{ todo["votes"] }}</td>

            <td class="gross">{{ todo["gross"] }}</td>

            <td class="func1"><a href="./remove?\_id={{ todo['\_id'] }}"><button type="submit">DELETE</button></a></td>

            <td class="func1"><a href="./update?\_id={{ todo['\_id'] }}"><button type="submit">EDIT</button></a></td>

        </tr>

        {% endfor %}

    {% else %}

        <h4>No Result Found !!</h4>

    {% endif %}

    </table>

</hr>

</br>

    <a href="/list">Return to TaskList</a>

</body>

</html>

Front-end: update.html

<html>

    <head>

        <title>{{t}}</title>

    </head>

    <style>

        h1{

            font-family:"Arial Black", Gadget, sans-serif;

        }

        body{

            background-color:rgb(235, 225, 225);

        }

    </style>

<body>

    <h1>{{h}}</h1>

    <hr>

    <h3>Update tasks with a reference</h3>

    <form action="/action3" method="POST">

    {% for task in tasks %}

        Unique Object ID : {{ task['\_id'] }}<br/>

    <input type="text" name="\_id" value="{{ task['\_id'] }}" hidden>

    <table>

    <tr>

    <td>name</td><td> : </td><td><input type="text" name="name" value="{{ task['name'] }}" placeholder="{{ task['name'] }}"></td>

    </tr>

    <tr>

    <td>year</td><td> : </td><td><input type="text" name="year" vlaue="{{ task['year'] }}" placeholder="{{ task['year'] }}"> </td>

    </tr>

    <tr>

    <td>runtime</td><td> : </td><td><input type="text" name="runtime" value="{{ task['runtime'] }}" placeholder="{{ task['runtime'] }}"></td>

    </tr>

    <tr>

    <td>genre</td><td> : </td><td><input type="text" name="genre" value="{{ task['genre'] }}" placeholder="{{ task['genre'] }}"></td>

    </tr>

    <tr>

        <td>rating</td><td> : </td><td><input type="text" name="rating" value="{{ task['rating'] }}" placeholder="{{ task['rating'] }}"></td>

        </tr>

    <tr>

        <td>metascore</td><td> : </td><td><input type="text" name="metascore" value="{{ task['metascore'] }}" placeholder="{{ task['metascore'] }}"></td>

    </tr>

    <tr>

        <td>timeline</td><td> : </td><td><input type="text" name="timeline" value="{{ task['timeline'] }}" placeholder="{{ task['timeline'] }}"></td>

    </tr>

    <tr>

        <td>votes</td><td> : </td><td><input type="text" name="votes" value="{{ task['votes'] }}" placeholder="{{ task['votes'] }}"></td>

    </tr>

    <tr>

        <td>gross</td><td> : </td><td><input type="text" name="gross" value="{{ task['gross'] }}" placeholder="{{ task['gross'] }}"></td>

    </tr>

    </table>

    {% endfor %}

        <button type="submit"> Update Task </button>

        <br/>

    </form>

    <a href="/list">List</a>

</body>

</html>

Back-end(python) : nosql.py

from flask import Flask, render\_template,request,redirect,url\_for # For flask implementation

from bson import ObjectId # For ObjectId to work

from pymongo import MongoClient

import os

app = Flask(\_\_name\_\_)

title = "NoSQL Project Application"

heading = "Movie-overview dataset"

client = MongoClient("mongodb://127.0.0.1:27017") #host uri

db = client.movies #Select the database

todos = db.review #Select the collection name

def redirect\_url():

return request.args.get('next') or \

request.referrer or \

url\_for('index')

@app.route("/list")

def lists ():

#Display the all Tasks

todos\_l = todos.find()

a1="active"

return render\_template('index.html',a1=a1,todos=todos\_l,t=title,h=heading)

@app.route("/")

@app.route("/uncompleted")

def tasks ():

#Display the Uncompleted Tasks

todos\_l = todos.find({"done":"no"})

a2="active"

return render\_template('index.html',a2=a2,todos=todos\_l,t=title,h=heading)

@app.route("/completed")

def completed ():

#Display the Completed Tasks

todos\_l = todos.find({"done":"yes"})

a3="active"

return render\_template('index.html',a3=a3,todos=todos\_l,t=title,h=heading)

@app.route("/done")

def done ():

#Done-or-not ICON

id=request.values.get("\_id")

task=todos.find({"\_id":ObjectId(id)})

if(task[0]["done"]=="yes"):

todos.update({"\_id":ObjectId(id)}, {"$set": {"done":"no"}})

else:

todos.update({"\_id":ObjectId(id)}, {"$set": {"done":"yes"}})

redir=redirect\_url()

return redirect(redir)

@app.route("/action", methods=['POST'])

def action ():

#Adding a Task

name=request.values.get("name")

year=request.values.get("year")

runtime=request.values.get("runtime")

genre=request.values.get("genre")

rating=request.values.get("rating")

metascore=request.values.get("metascore")

timeline=request.values.get("timeline")

votes=request.values.get("votes")

gross=request.values.get("gross")

todos.insert({ "name":name, "year":year, "runtime":runtime, "genre":genre, "rating":rating, "metascore":metascore, "timeline":timeline, "votes":votes, "gross":gross})

return redirect("/list")

@app.route("/remove")

def remove ():

#Deleting a Task with various references

key=request.values.get("\_id")

todos.remove({"\_id":ObjectId(key)})

return redirect("/")

@app.route("/update")

def update ():

id=request.values.get("\_id")

task=todos.find({"\_id":ObjectId(id)})

return render\_template('update.html',tasks=task,h=heading,t=title)

@app.route("/action3", methods=['POST'])

def action3 ():

#Updating a Task with various references

name=request.values.get("name")

year=request.values.get("year")

runtime=request.values.get("runtime")

genre=request.values.get("genre")

rating=request.values.get("rating")

metascore=request.values.get("metascore")

timeline=request.values.get("timeline")

votes=request.values.get("votes")

gross=request.values.get("gross")

id=request.values.get("\_id")

todos.update({"\_id":ObjectId(id)}, {'$set':{ "name":name, "year":year, "runtime":runtime, "genre":genre, "rating":rating, "metascore":metascore, "timeline":timeline, "votes":votes, "gross":gross }})

return redirect("/")

@app.route("/search", methods=['GET'])

def search():

#Searching a Task with various references

key=request.values.get("key")

refer=request.values.get("refer")

if(key=="\_id"):

todos\_l = todos.find({refer:ObjectId(key)})

else:

todos\_l = todos.find({refer:key})

return render\_template('searchlist.html',todos=todos\_l,t=title,h=heading)

if \_\_name\_\_ == "\_\_main\_\_":

app.run()

**CHAPTER 7**

### REFERENCES

1. <https://data.world/datasets/food>
2. <https://realpython.com/tutorials/flask/>
3. <https://docs.mongodb.com/manual/core/crud/>
4. <https://www.mongodb.com/blog/post/getting-started-with-python-and-mongodb>
5. <https://opensource.com/article/18/4/flask>
6. <https://flask.palletsprojects.com/en/2.0.x/>
7. https://fdc.nal.usda.gov/download-datasets.html