**Mern Stack Journal**

**Name: Khushi Jaydeep Shah**

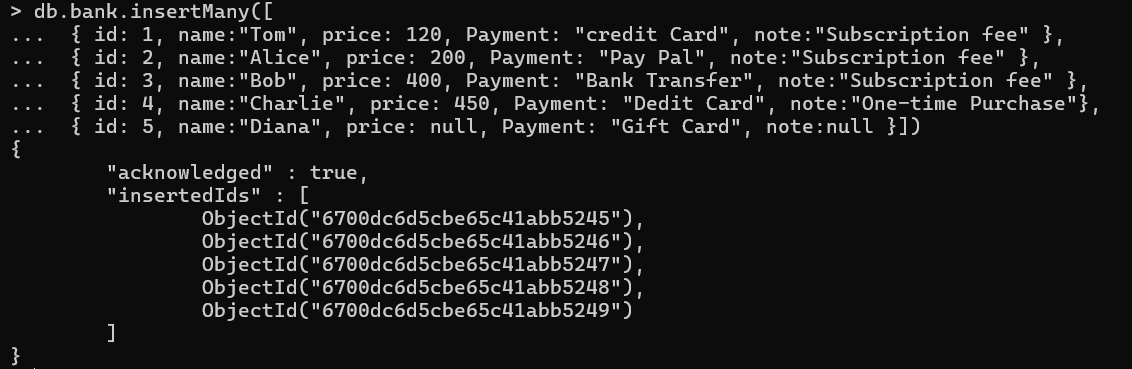
**Seat No: 31011222061**

**Roll No: 37**

**Class: TYBCA**

# *Practical-1*

1. Create a database bank id name transaction with an array with attributes id , price, Subscriber payment and note. Insert at least five records and perform following operations:



* Find any record where the name is Tom.



* Find any record where the total payment amount is 400.



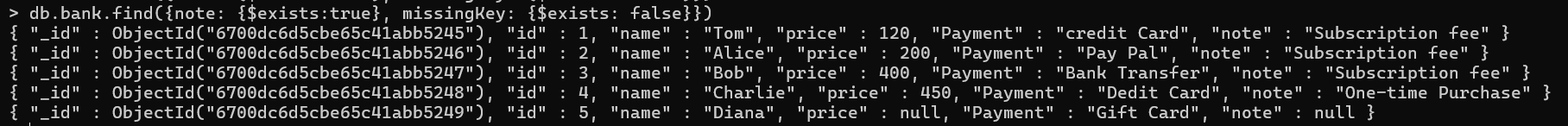
* Find any record where the price is greater than 400.



* Find any record where the note is null or the key itself is missing.



* Find any record where note exist and key is missing.



* Find any record where the note and key does not exist.

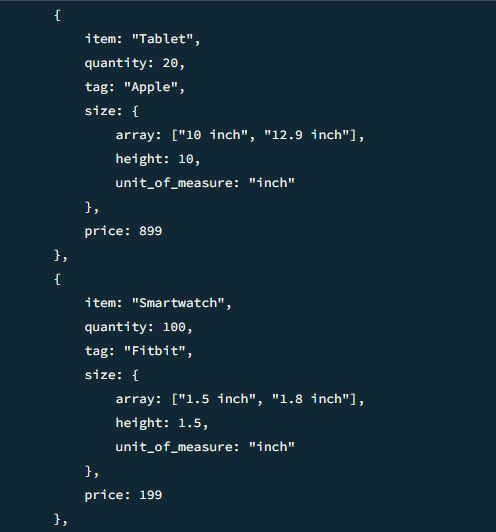


# 

# *Practical-2(05-08-24)*

1. **Write a Mongodb query for product with fields item name, quantity, tag, size (with array, height, with unit of measure) and price of product**

* Insert at least five records.





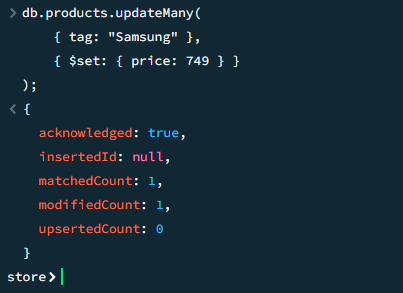
* Display records in pretty format



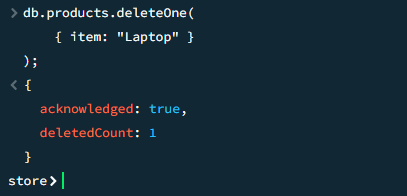




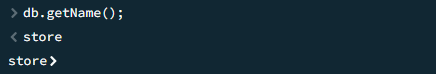
* Update record of price where product tag is Samsung.



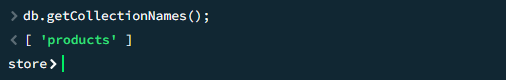
* Delete record of product where the item is laptop.



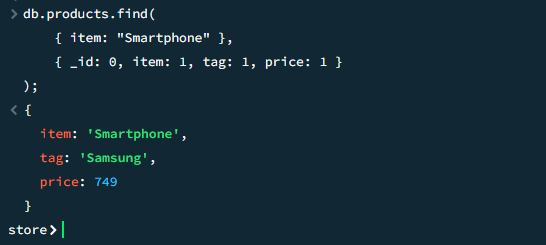
* Display current database name



* Display current collection name



* Display record of product with item name mobile and display only item tag and price..

.

# *Practical-3(05-08-24)*

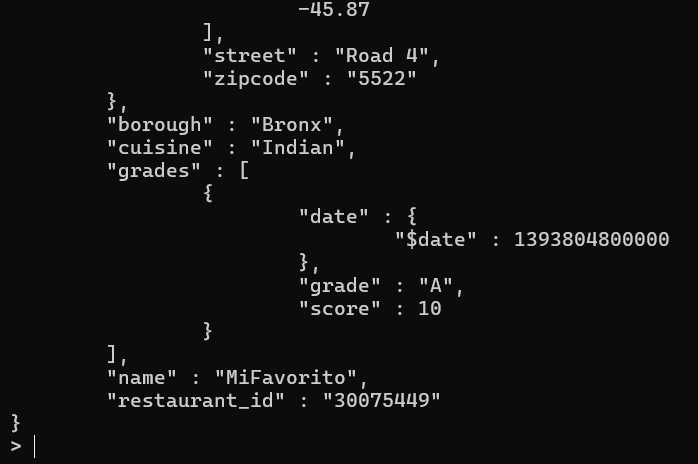
1.Perform the following queries on collection restaurant

* Write a Mongodb query to display all the documents from the collection restaurant.

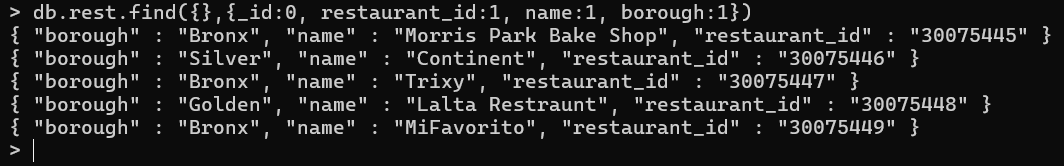




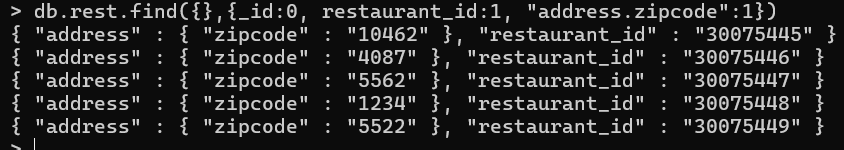




* Write a Mongodb query to display field restaurant id, name and borough for all the documents from collection restaurant



* Write a mongo db query to display field restaurant id and zip code, but exclude field\_id for all the documents from the collection restaurant



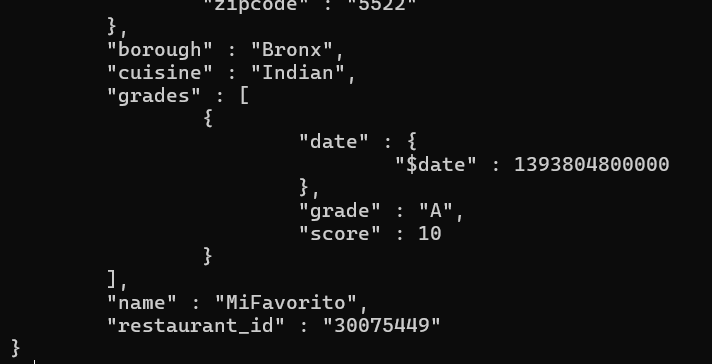
* Write a query to display first two restaurants, which is in Borough Bronx





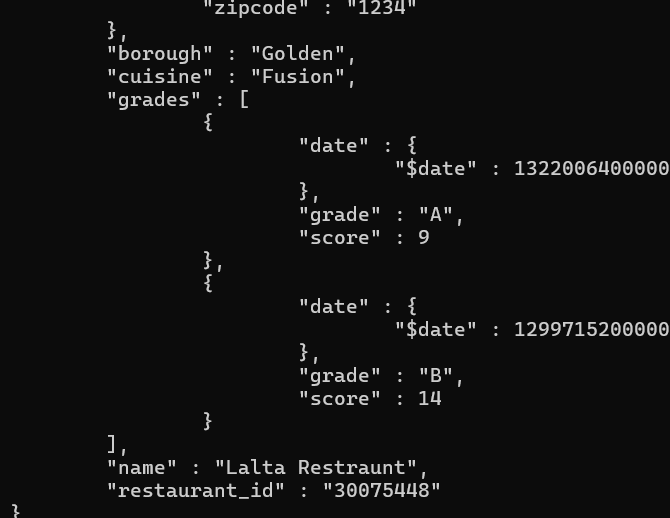
* Write a Mongodb query to display the next two restaurants after skipping 1st which are in borough Bronx





* Writer Mongodb query to find the restaurant which achieved a score more than 10





* Write a Mongodb query to find the restaurants that achieved a score more than 5 but less than 10



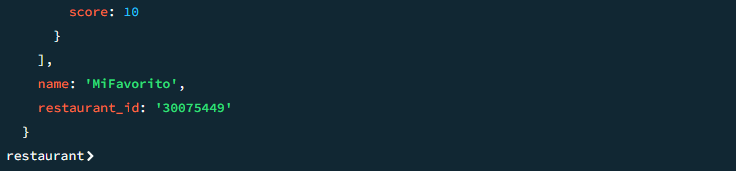




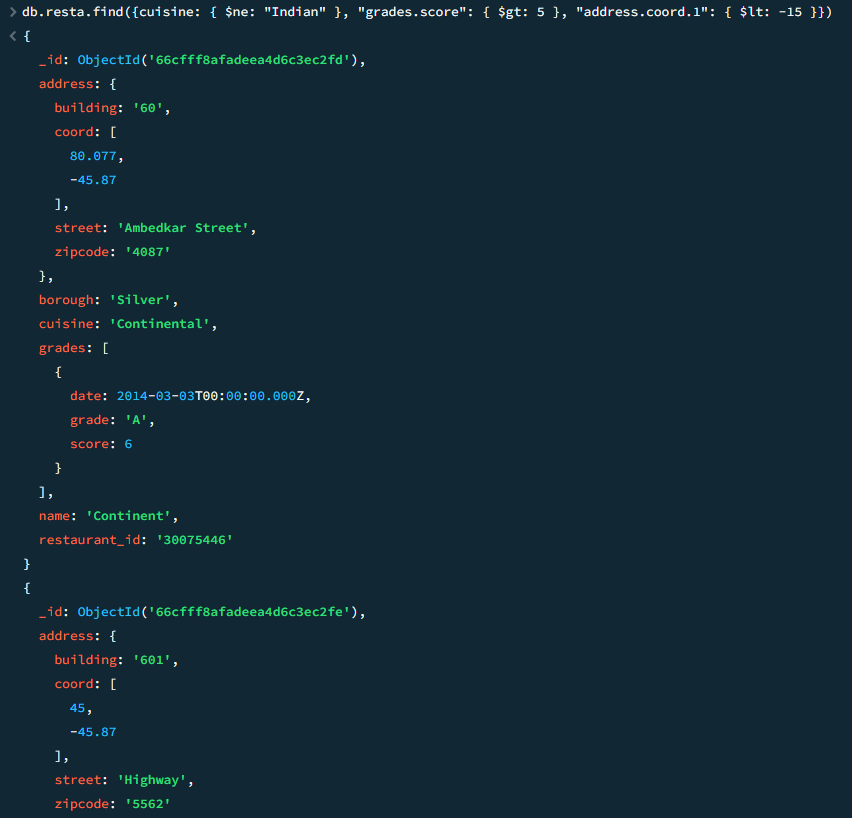
* Write a Mongodb query to find the restaurants which locate in latitude value less than -15

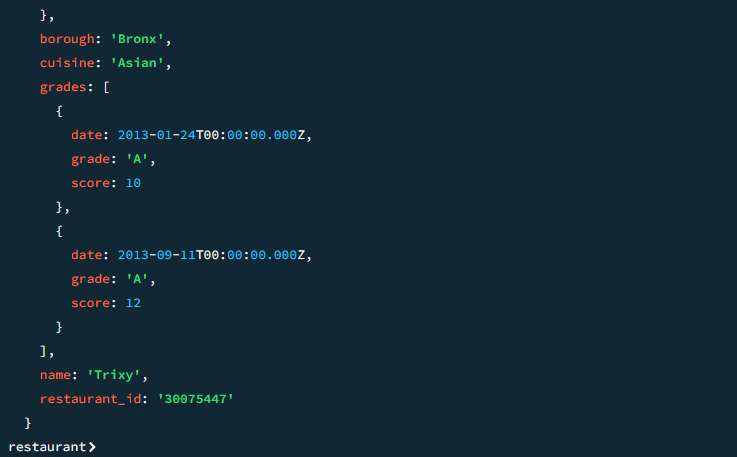




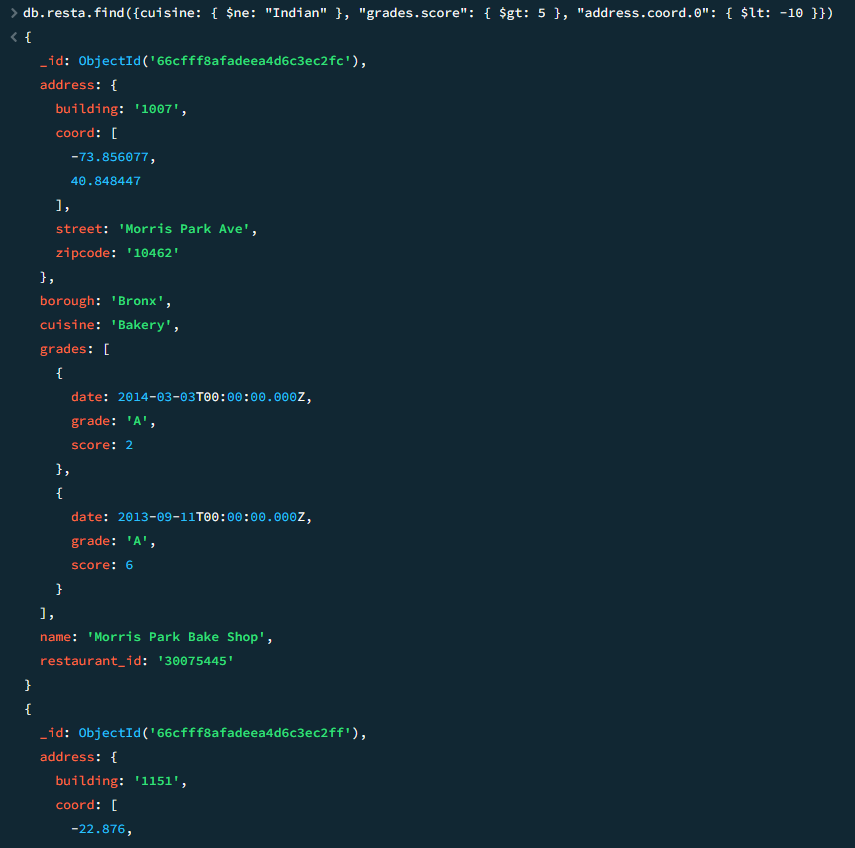


* Write a Mongodb query to find the restaurants that do not prepare any cuisine of “Bakery” and their grade score is more than 5, and latitude less than -10



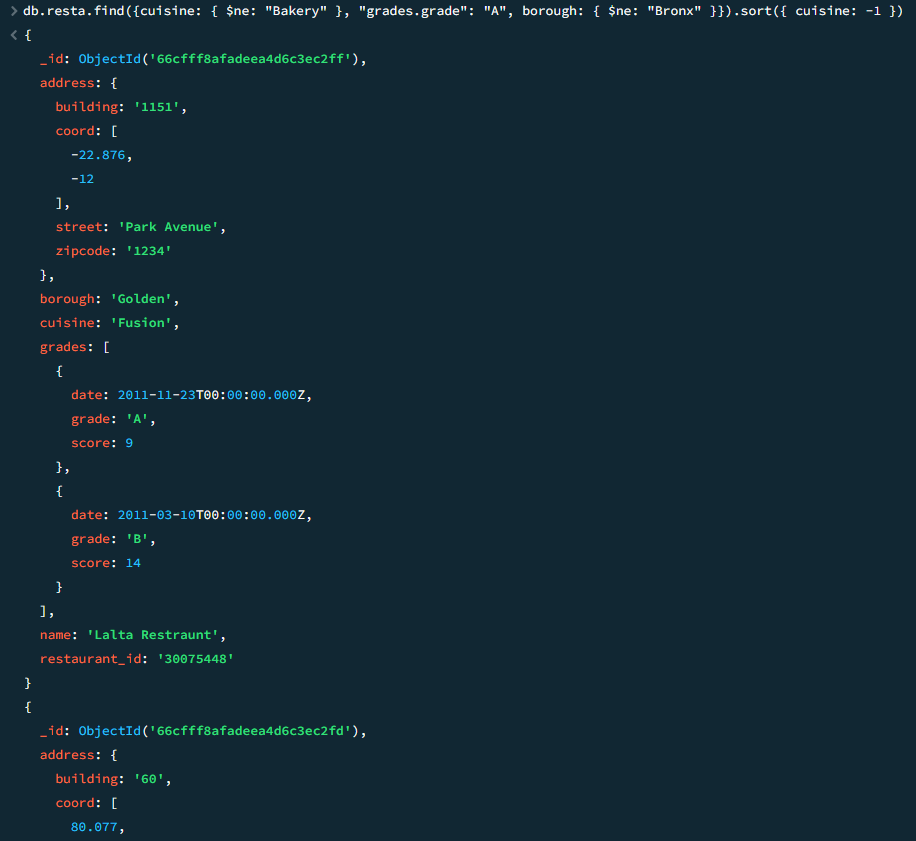


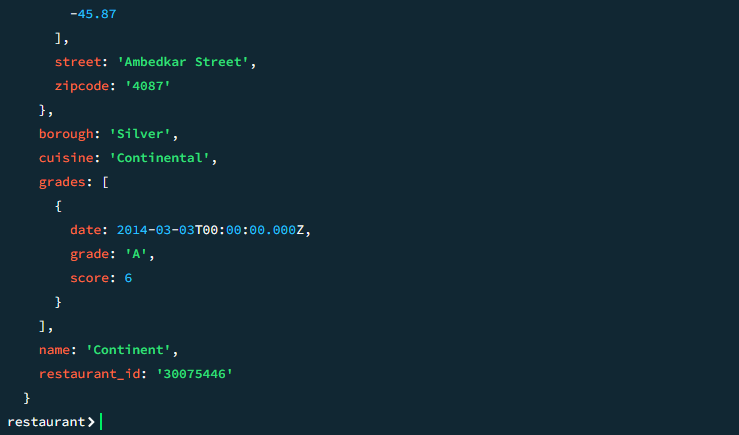
* Write a Mongodb query to find the restaurant which does not prepare any cuisine of “Bakery” and achieved a score of more than 5 and located in the longitude of less than -10



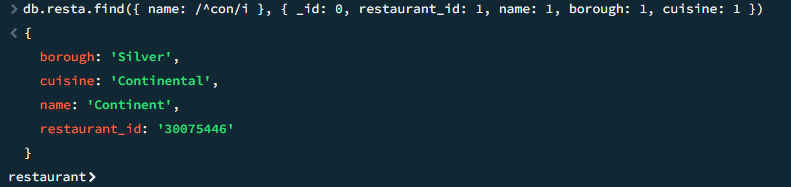


* Write a Mongodb query to find the restaurants which do not prepare any cuisine of “Bakery” and achieved a grade point ‘A’, does not belong to the Borough Bronx. The document must be displayed according to the cuisine in descending order

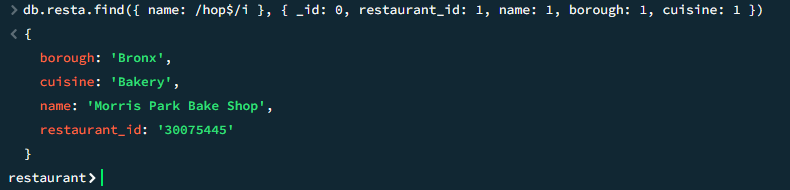




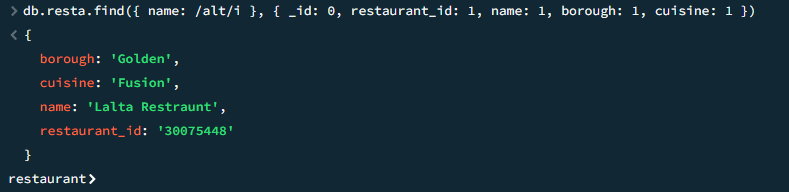
* Write a Mongodb query to find the restaurant id, name, borough and cuisine for those restaurants which contain “con” as first 3 letters of its name



* Write a Mongodb query to find the restaurant id, name, borough and cuisine for those restaurants which contain “hop” as last 3 letters of its name



* Write a Mongodb query to find the restaurant id, name, borough and cuisine for those restaurants which contain “alt” as 3 letters somewhere in its name



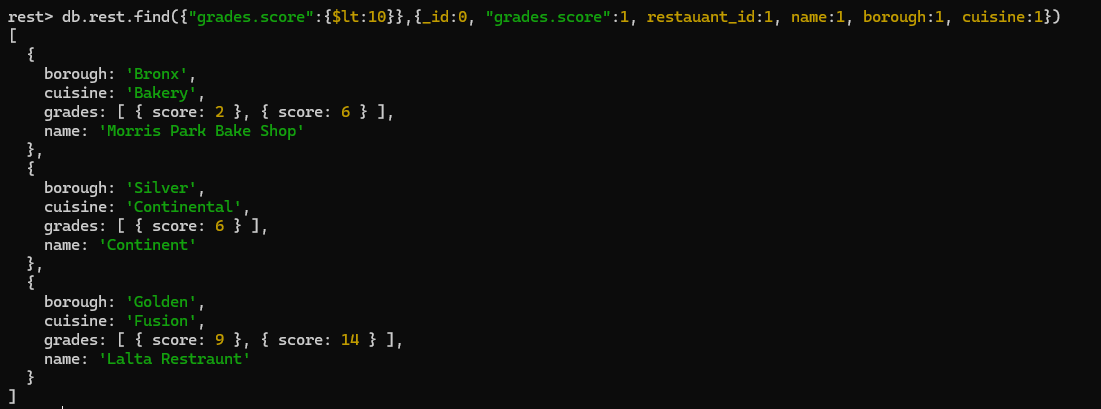
* Write a Mongodb query to find the restaurants which belong to the Bronx and prepare either Indian or Asian Cuisine.



* Write a Mongodb query to find the restaurant id, name, borough and cuisine for those restaurants which belong to Golden or Bronx.



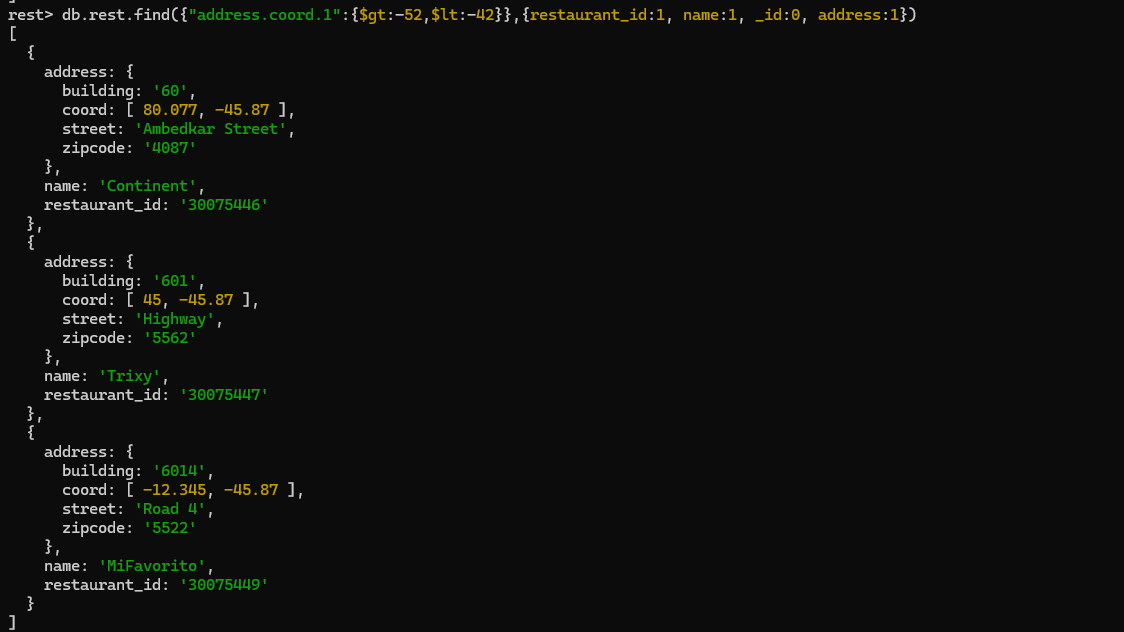
* Write a Mongodb query to find the restaurant id, name, borough and cuisine for those restaurants whose achieved score is not more than 10.



* Write a Mongodb query to find the restaurant id, name, borough and cuisine for those restaurants which prepare cuisine except Asian and Chinese or restaurants name begins with letters “Mi”.



* Write a Mongodb query to find the restaurant id, name, address and geographical location for those restaurants where the second element of the coords array contains a value which is more than -52 and less than -42.



* Write a Mongodb query to arrange the name of the restaurants in ascending order along with all the columns.



* Write a Mongodb query to arrange the name of the restaurants in descending order along with all the columns.



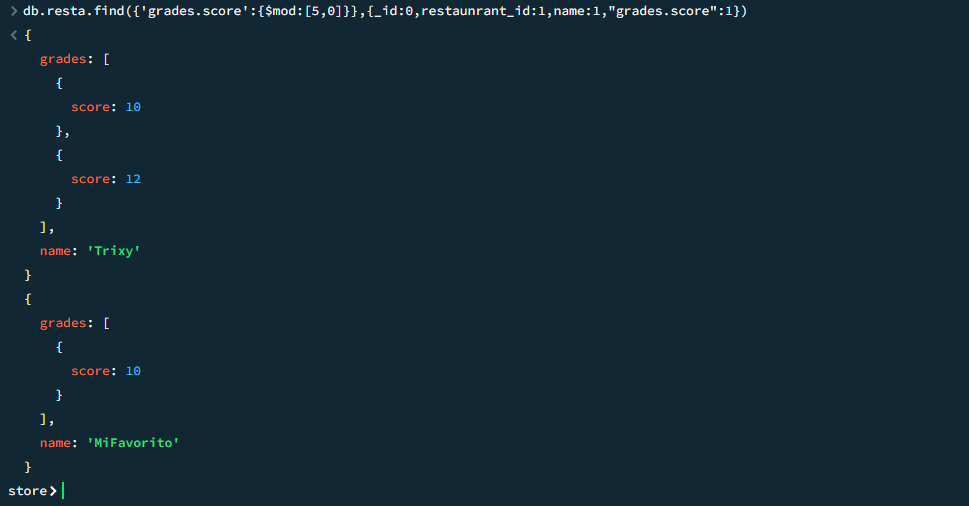
* Write a Mongodb query to know whether all the addresses contain the street or not



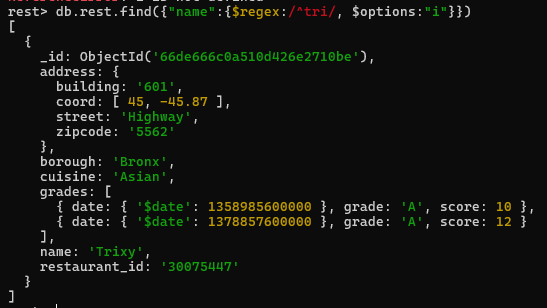
* Write a Mongodb query which will select all the documents in the restaurant collection where the coord field is double.



* Write a Mongodb query which will display restaurant\_id, name and grade for those restaurants which returns 0 after diving score by 5.



* Write a Mongodb query which will display name, borough and coords for those restaurants which contain ‘tri’ as first three letters of its name.



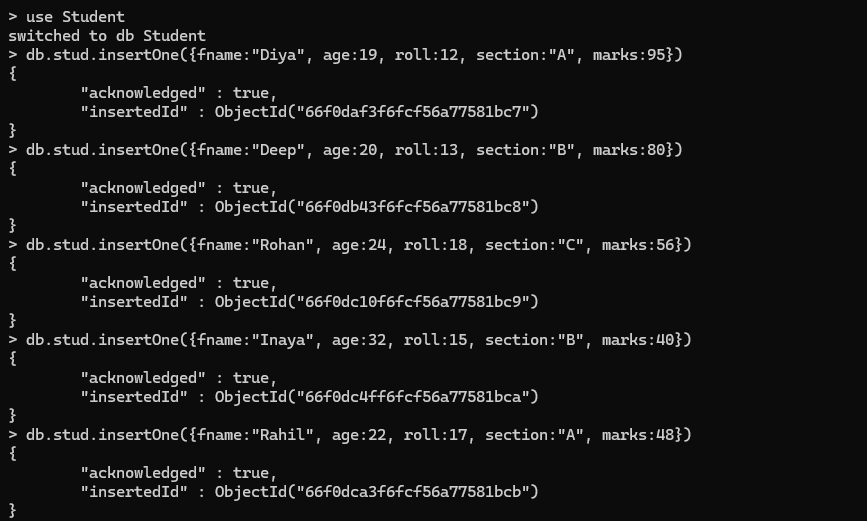
* Write a Mongodb query which will display name, borough and coords for those restaurants which contain ‘vor’ as three letters somewhere in its name.



# *Practical - 4 (23-09-24)*

Create collection for student with attributes name,age,roll no, sectio(a,b,c), marks(out of 100)

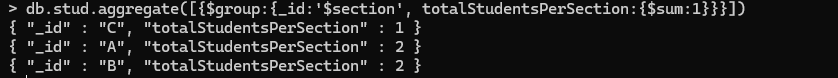
Insert at least 5 docs



Display total no of students from sec B



Display total no of students from section vise



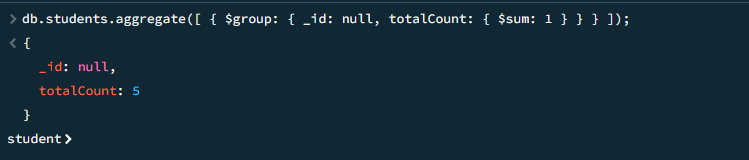
Display max age



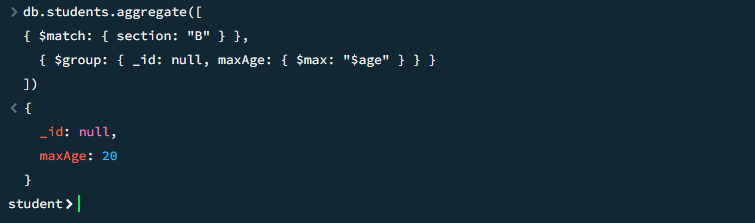
Display details of student whose age is greater than 30



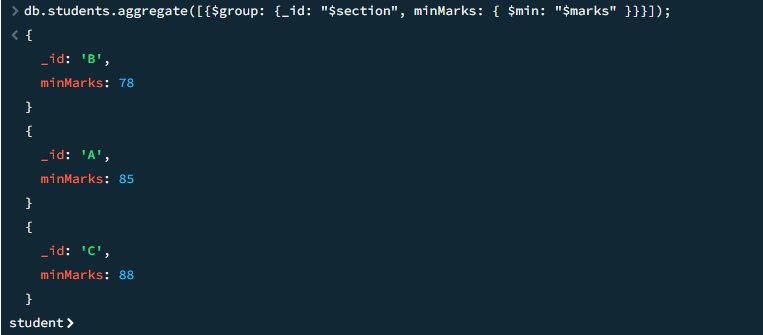
Count total no of docs



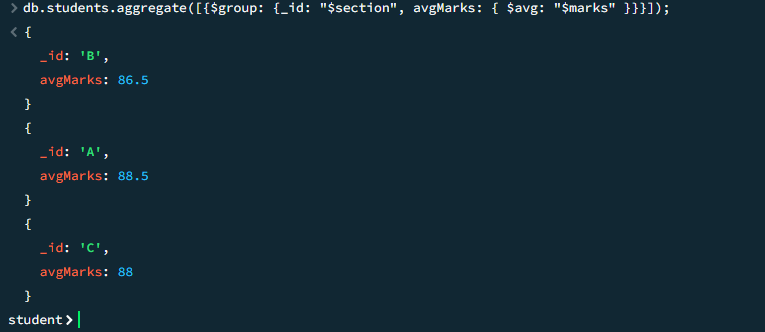
Display details of students having max age in section B



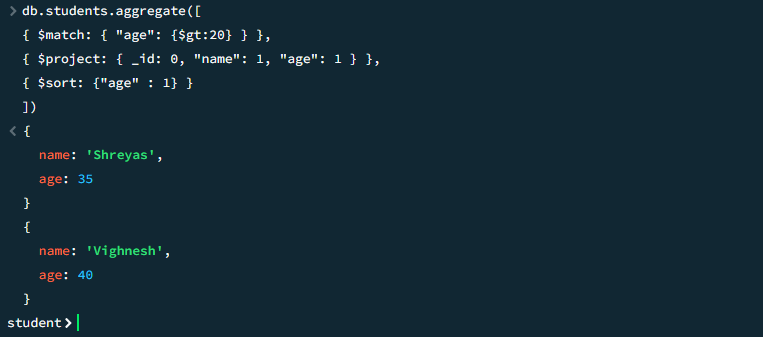
Display min marks of student group by section



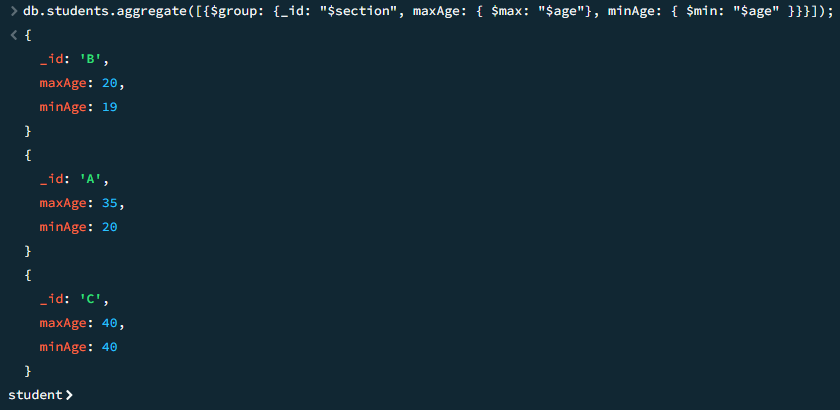
Display avg marks of student group by section



Find the student whose age is greater than 20 and display only name and age of that student also sort the data acc to age in ascending order



Display details of student group by section with max age and min age(id, section, min age, max age)



Demonstrate mongodb import and export collection

# 

# *Practical-5 (12-08-24)*

1.WAP in react js to display your name with some greeting msg, display text in center and red in color

app.js

import logo from './logo.svg';

import './App.css';

function App() {

return (

<div className="App">

<p style={{color:"red"}}>Hello Khushi</p>

</div>

);

}

export default App;

Output:



2.write a prog to create functional component called my info which contains ur name in h1 tag and red in color, with center aligned, use ordered list to display the list of top 3 vacation spots blue in color , use either internal or external css

app.js

import logo from './logo.svg';

import './App.css';

function myInfo() {

return (

<div className="App">

<h1 className='name'>Hello Khushi</h1>

<ol className='vacation'>

<li>Lonavala</li>

<li>Matheran</li>

<li>Igatpuri</li>

</ol>

</div>

);

}

export default myInfo;

Index.css

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

.name{

color: red;

text-align: center;

}

.vacation{

display: inline-block;

color: blue;

}

Output



# *Practical - 6*

1.WAP in react js to create app with functional component to display info of emp(empname, empid, depname and sal)use props to display atleast 5 emp info

index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import EmpInfo from './Emp.jsx';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<>

<h1 style={{color:'red', textAlign:'center'}}>Employee's Data</h1>

<EmpInfo ename="Khushi" id="1" dname="IT" sal="15000"/>

<EmpInfo ename="Dhruti" id="2" dname="HR" sal="14000"/>

<EmpInfo ename="Hiya" id="3" dname="IT" sal="10000"/>

<EmpInfo ename="Drashtii" id="4" dname="MG" sal="12000"/>

<EmpInfo ename="Ria" id="5" dname="ACCOUNTS" sal="18000"/>

</>

)

Index.css

body {

margin: 0;

font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

code {

font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

monospace;

}

.name{

color: red;

text-align: center;

}

.vacation{

display: inline-block;

color: blue;

}

.emp{

text-align: center;

}

Emp.jsx

function EmpInfo(props)

{

return (

<div className="emp">

<h1>Employee {props.id}</h1>

<p> Employee {props.ename} with id: {props.id} from department: {props.dname} has salary: {props.sal}</p>

<hr></hr>

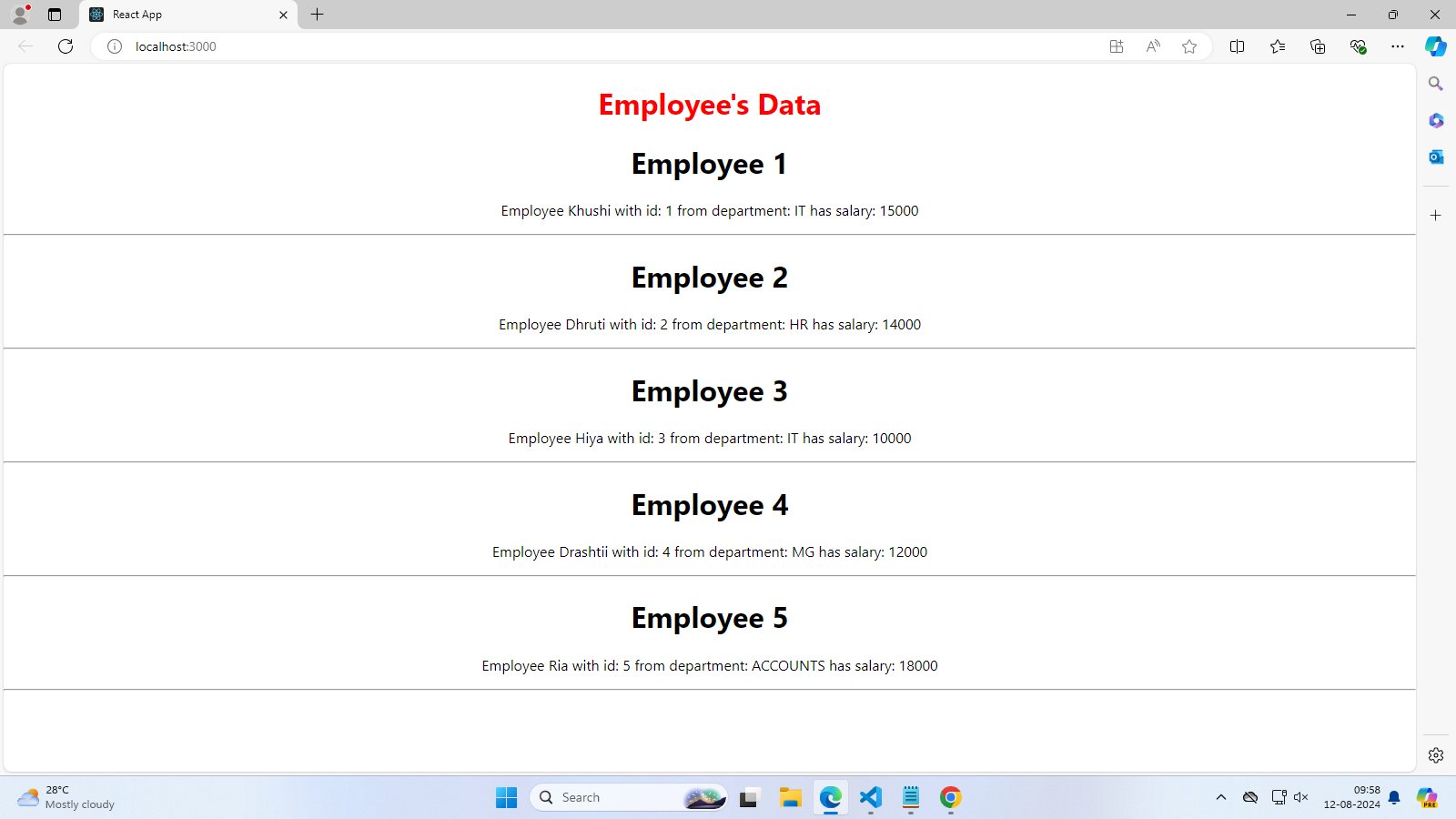
</div>

);

}

export default EmpInfo;

Output



2.WAP in react js to create app with functional component to display info of student(name id and contact no) display at least 5 student info using props and display the data in table format

1. App.js

function Stud(props) {

return(

<>

<tr>

<td>{props.id}</td>

<td>{props.name}</td>

<td>{props.contact}</td>

</tr>

</>

)

}

export default Stud;

1. index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import reportWebVitals from './reportWebVitals';

import Stud from './App'

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<Stud/>

<h1 style={{color: 'red', textAlign: 'center'}}>Student's Data</h1>

<center>

<table>

<tr>

<th>Student Id</th>

<th>Student Name</th>

<th>Student Contact</th>

</tr>

<Stud name='Smit' id='1' contact='1000000000' />

<Stud name='Sahil' id='2' contact='2000000000' />

<Stud name='Shreyas' id='3' contact='3000000000' />

<Stud name='Vighnesh' id='4' contact='4000000000' />

<Stud name='Harsh' id='5' contact='5000000000' />

</table>

</center>

</React.StrictMode>

);

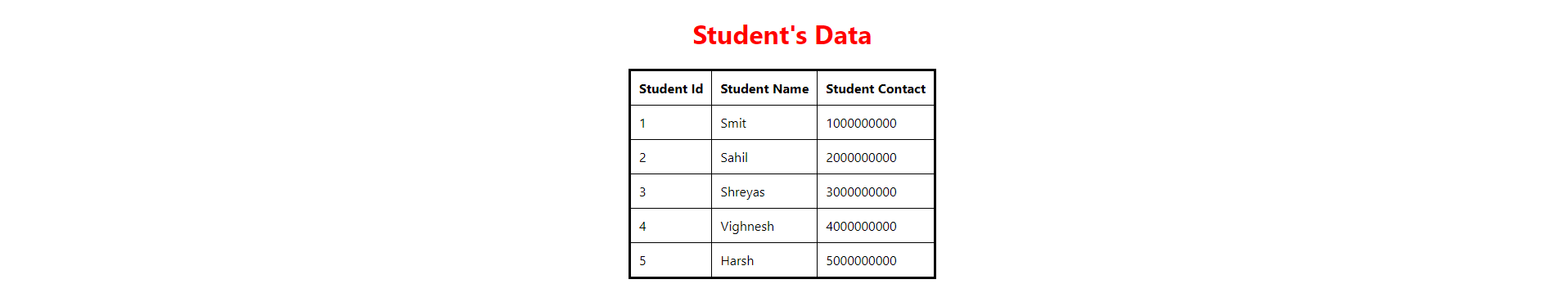
// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

Output:



# *Practical-7*

1.Write a program in react js to create class component to display product cart page

Card.jsx:

import React, { Component } from 'react';

import './card.css';

import logo from './logo.svg';

class Card extends Component {

state = {

cards: [

{ img: logo, brand: 'Dior', price: '5000' },

{ img: logo, brand: 'Chanel', price: '6000' },

{ img: logo, brand: 'Gucci', price: '7000' },

{ img: logo, brand: 'Prada', price: '8000' }

]

};

render() {

return (

<div className="card-pract-container">

{

this.state.cards.map((card, index) => (

<div className="card1" key={index}>

<img src={card.img} alt={card.brand}/>

<h2>Brand: {card.brand}</h2>

<p>Price: {card.price} rupees</p>

</div>

))}

</div>

);

}

}

export default Card;

1. index.js:

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import reportWebVitals from './reportWebVitals';

import Card from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<Card />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

1. card.css:

body{

margin: 0 !important;

padding: 0 !important;

display: flex !important;

flex-direction: column !important;

justify-content: center !important;

align-items: center !important;

}

.card1{

width: 18rem;

height: 20rem;

background-color: orange;

border-radius: 3rem;

padding: 5px;

align-items: center;

text-align: center;

justify-content: center;

margin-left: 5rem;

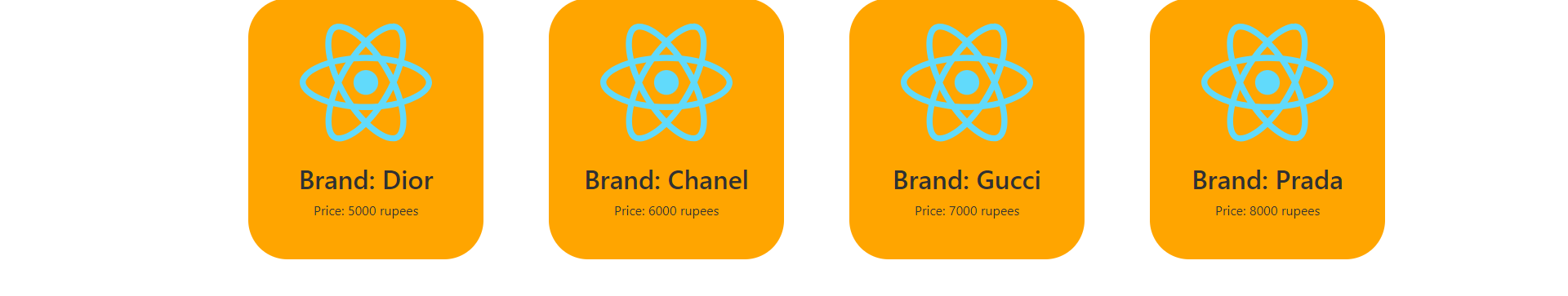
}

.card-pract-container{

display: flex;

}

Output:



# *Practical-8(26-08-24)*

1.use class component to perform following

a) increment number of like when u click

b) display small 's' of likes only when the no is greater than or = 2

Code

index.js

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import Component from './App.js';

ReactDOM.render(

<div>

<Component/>

</div>,document.getElementById('root')

);

app.js

import React from 'react';

class Component extends React.Component {

state = {

count:0

}

render() {

return(

<>

<h1>You have received {this.state.count} {this.state.count >= 2? "likes":"like"}</h1>

<button onClick={this.incLikes}>Like</button>

</>

)

}

incLikes = () => {

this.setState({

count:this.state.count + 1

})

}

}

export default Component;

Output





2.display digital clock using use effect(class component or function component)

1. Clock.jsx:

import React, { Component, useEffect, useState } from 'react';

export default function Clock() {

let d1 = new Date().toLocaleTimeString()

const [d, setDatet] = useState(d1)

useEffect( () => {

setTimeout( () =>{

setDatet(new Date().toLocaleTimeString())

}, 1000);

}

)

return(

<>

<h1>Today's Date is {d}</h1>

</>

)

}

1. index.jsx:

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import reportWebVitals from './reportWebVitals';

import Clock from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<Clock />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

Output:

****

# *Practical - 9*

1. **WAP in React.js to add state to stateless functional component. State should have a property that is logged in which is boolean. Render if the user is logged in or logged out.**

App.jsx:  
import React, { useState } from 'react';

import './App.css'

function App() {

const [isLoggedIn, setIsLoggedIn] = useState(false);

const toggleLogin = () => {

setIsLoggedIn(prevState => !prevState);

};

return (

<div style={{ textAlign: 'center', marginTop: '50px' }}>

<h1>{isLoggedIn ? 'Welcome Back!' : 'Please Log In'}</h1>

<button onClick={toggleLogin}>

{isLoggedIn ? 'Logout' : 'Login'}

</button>

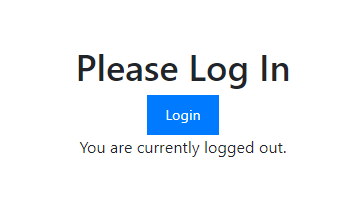
<p>You are currently {isLoggedIn ? 'logged in' : 'logged out'}.</p>

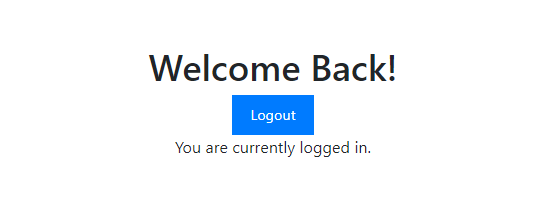
</div>

);

}

export default App;

Output:  
 



1. **WAP in React.js to demonstrate the use of array Cube as state variable. Display a cube of numbers on every single click.**

App.jsx:

import React, { useState } from 'react';

import './App.css'

function App() {

const [cube, setCube] = useState([]);

const handleClick = () => {

const nextNumber = cube.length;

const nextCube = nextNumber \*\* 3;

setCube([...cube, nextCube]);

};

return (

<div>

<h1>Cube Calculator</h1>

<button onClick={handleClick}>Calculate Cube</button>

<ul>

{cube.map((value, index) => (

<li key={index}>Cube of {index} is {value}.</li>

))}

</ul>

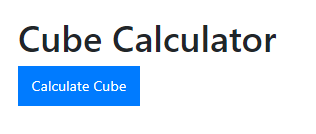
</div>

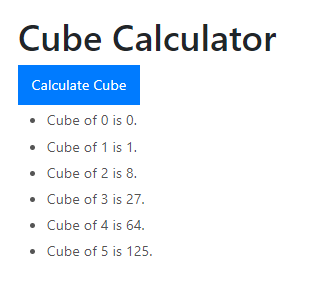
);

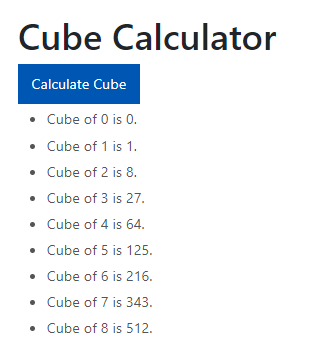
};

export default App;

Output:







# *Practical - 10*

1. **Basic Calculator:**

Calculator.jsx:

import React, { useState } from 'react';

function App() {

const [num1, setNum1] = useState(0);

const [num2, setNum2] = useState(0);

const [operator, setOperator] = useState('+');

let ans = 0;

const incrementNum1 = () => {

setNum1(prevValue => prevValue + 1);

};

const incrementNum2 = () => {

setNum2(prevValue => prevValue + 1);

};

const handleOperatorChange = (e) => {

setOperator(e.target.value);

};

const calculate = () => {

switch (operator) {

case '+':

return ans = num1 + num2;

case '-':

return ans = num1 - num2;

case '\*':

return ans = num1 \* num2;

case '/':

return ans = num1 / num2;

default:

return ans;

}

};

return (

<div style={{ textAlign: 'center', marginTop: '50px' }}>

<h1>Basic Calculator</h1>

<button onClick={incrementNum1}>{num1}</button>

<select onChange={handleOperatorChange} value={operator}>

<option value="+">+</option>

<option value="-">-</option>

<option value="\*">\*</option>

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

</select>

<button onClick={incrementNum2}>{num2}</button>

<h3>Result: {calculate()}</h3>

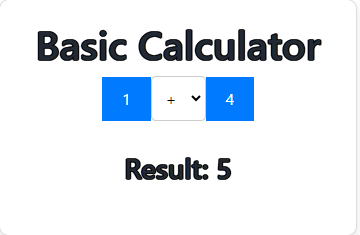
</div>

);

}

export default App;

Output:



1. **WAP in REACT.js to accept name, mobile and languages known from user onclick of submit button. Display all info on same page. User spread operator. Use checkbox for languages.**

App.jsx:

import React, { useState } from 'react';

import './App.css';

const App = () => {

const [formData, setFormData] = useState({

name: '',

mobile: '',

languages: {

English: false,

Hindi: false,

Marathi: false,

German: false,

Japanese: false,

French: false,

},

});

const [submittedData, setSubmittedData] = useState(null);

const handleChange = (e) => {

const { name, value, type, checked } = e.target;

if (type === 'checkbox') {

setFormData((prevData) => ({

...prevData,

languages: {

...prevData.languages,

[name]: checked,

},

}));

} else {

setFormData((prevData) => ({

...prevData,

[name]: value,

}));

}

};

const handleSubmit = (e) => {

e.preventDefault();

setSubmittedData(formData);

};

return (

<div>

<h1>User Information Form</h1>

<form onSubmit={handleSubmit}>

<div>

<label>

Name:

<input

type="text"

name="name"

value={formData.name}

onChange={handleChange}

required

/>

</label>

</div>

<div>

<label>

Mobile:

<input

type="tel"

name="mobile"

value={formData.mobile}

onChange={handleChange}

required

/>

</label>

</div>

<div>

<h3>Languages Known:</h3>

{Object.keys(formData.languages).map((language) => (

<label key={language}>

<input

type="checkbox"

name={language}

checked={formData.languages[language]}

onChange={handleChange}

/>

{language}<br />

</label>

))}

</div>

<button type="submit">Submit</button>

</form>

{submittedData && (

<div>

<h2>Submitted Information</h2>

<p>Name: {submittedData.name}</p>

<p>Mobile: {submittedData.mobile}</p>

<p>Languages Known:</p>

<ul>

{Object.keys(submittedData.languages).map((language) =>

submittedData.languages[language] ? (

<li key={language}>{language}</li>

) : null

)}

</ul>

</div>

)}

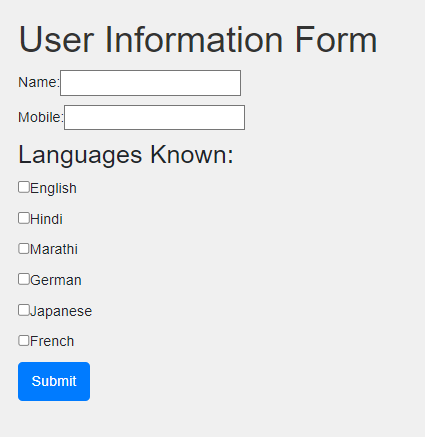
</div>

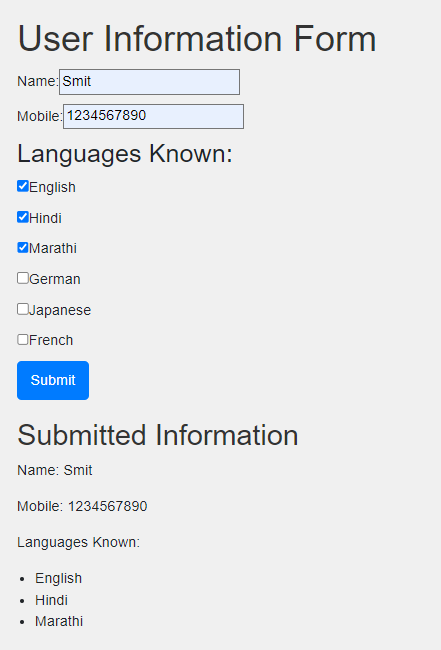
);

};

export default App;

Output:

’

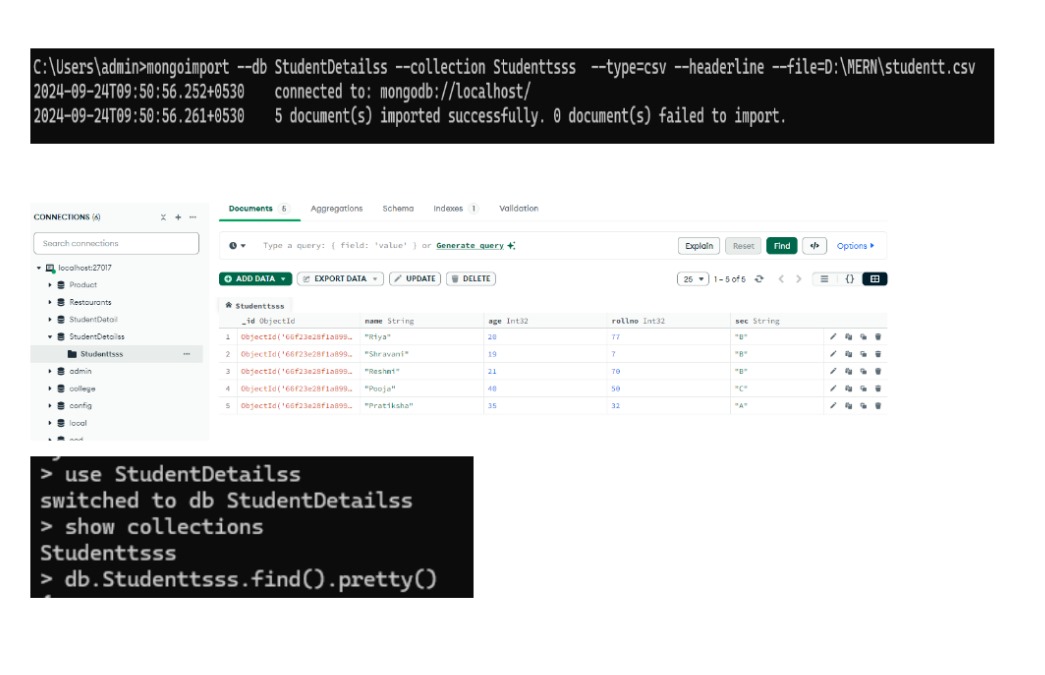


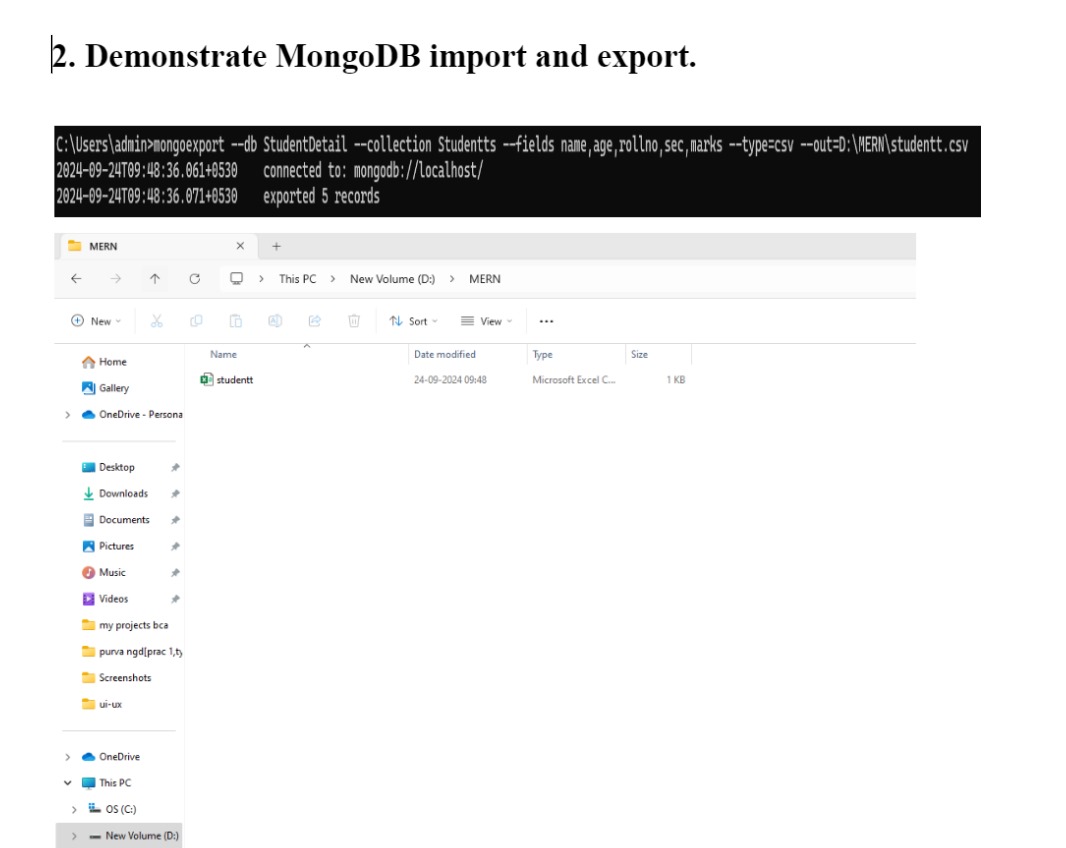
# *Practical - 11*

1. **Demonstrate the use of MongoDB EXPORT and IMPORT (.csv file)**

**Operation.**

**IMPORT:**

****

****

# *Practical - 12*

1. **WAP in React to write a program in Home Categories and About Us, in the category page create links for any 3 products. Perform Routing to create links between different pages.**

App.js:

import React from 'react';

import { BrowserRouter, Route, Routes } from 'react-router-dom';

import Home from './Home';

import Categories from './Categories';

import Product from './Product';

import AboutUs from './AboutUs';

const App = () => {

return (

<BrowserRouter>

<div>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/categories" element={<Categories />} />

<Route path="/product/:id" element={<Product />} />

<Route path="/aboutus" element={<AboutUs />} />

</Routes>

</div>

</BrowserRouter>

);

};

export default App;

Home.jsx:

import React from 'react';

import { Link } from 'react-router-dom';

import './Home.css'

const Home = () => {

return (

<div className='container'>

<Link to="/" className='link'>Home</Link>

<Link to="/categories" className='link'>Categories</Link>

<Link to="/aboutus" className='link'>About Us</Link>

<h1>Welcome to Shopping Mart</h1>

</div>

);

};

export default Home;

Home.css:

.container{

text-align: center;

}

.link{

margin: 20px;

}

Categories.jsx:

import React, { useState } from 'react';

import { Link } from 'react-router-dom';

import './Home.css';

const products = [

{ id: 1, name: "Product 1", description: "Description for Product 1" },

{ id: 2, name: "Product 2", description: "Description for Product 2" },

{ id: 3, name: "Product 3", description: "Description for Product 3" },

];

const Categories = () => {

const [selectedProduct, setSelectedProduct] = useState(null);

const handleProductClick = (product) => {

setSelectedProduct(product);

};

return (

<div className='container'>

<Link to="/" className='link'>Home</Link>

<Link to="/categories" className='link'>Categories</Link>

<Link to="/aboutus" className='link'>About Us</Link>

<h1>This is the Category Page</h1>

<div>

{products.map(product => (

<button key={product.id} onClick={() => handleProductClick(product)} className='link'>

{product.name}

</button>

))}

</div>

{selectedProduct ? (

<div className='product-details'>

<h2>{selectedProduct.name}</h2>

<p>{selectedProduct.description}</p>

</div>

) : (

<p>Select a product to see the details.</p>

)}

</div>

);

};

export default Categories;

AboutUs.jsx:

import React from 'react';

import { Link } from 'react-router-dom';

import './Home.css'

const Home = () => {

return (

<div className='container'>

<Link to="/" className='link'>Home</Link>

<Link to="/categories" className='link'>Categories</Link>

<Link to="/aboutus" className='link'>About Us</Link>

<h1>This is About Page</h1>

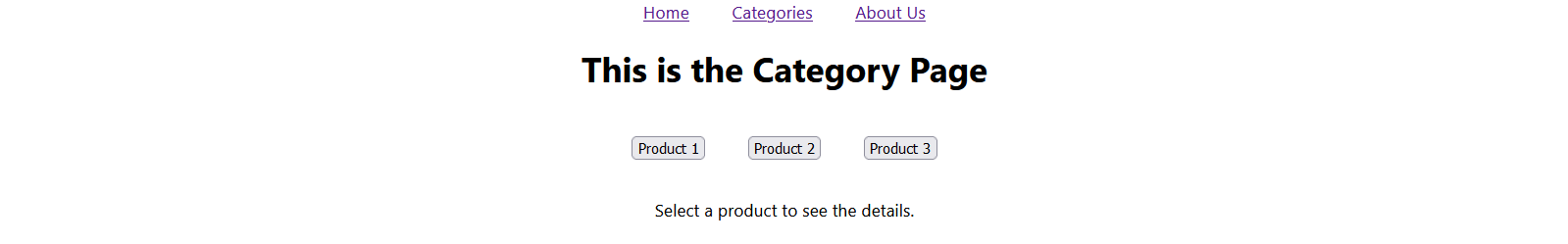
</div>

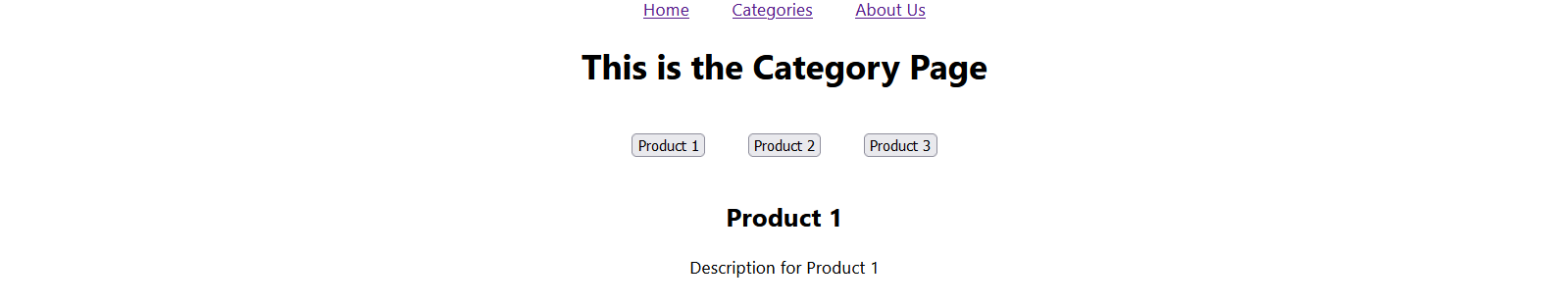
);

};

export default Home;

Output:  





–x–