------------------> Vegie.js

```
import React, {useState} from 'react'

const vegetableData =
{
    fruits: ['Tomato', 'Cucumber', 'Bell Pepper', 'Apple', 'Banana',
'Grapes'],
    leafyGreens: ['Spinach', 'Kale', 'Lettuce'],
    rootVegetables: ['Carrot', 'Potato', 'Beetroot'],
    legumes: ['Lentils', 'Chickpeas', 'Black Beans'],
};

export const Vegeis = () =>
{
    const [selectedCategory, setSelectedCategory] = useState('fruits');
    // const [errorMessage, setErrorMessage] = useState('');

    const handleCategoryChange = (e) =>
    {
        const category = e.target.value;
        setSelectedCategory(category);
    };

  return (
    <div className="vegetable-categories">
      <div className="category-dropdown">
        <label htmlFor="categorySelect">Select a Category:</label>
        <select id="categorySelect" value={selectedCategory}
onChange={handleCategoryChange}>
          <option value="Select Category" >Select</option>
          <option value="fruits">Fruits</option>
          <option value="leafyGreens">Leafy Greens</option>
          <option value="rootVegetables">Root Vegetables</option>
          <option value="legumes">Legumes</option>
          <option value="other">Other</option>
        </select>
      </div>
      <div className="vegetable-list">
        {selectedCategory === 'Select Category' ? (<p>Please Select A
Category</p>):
        vegetableData[selectedCategory] ? (
          <ul>
            {
              vegetableData[selectedCategory].map((veg,i) => (
                <li key={i}>{ veg }</li>
              ))
            }
          </ul>
        ): (<p>Category Not Found</p>)}
      </div>
    </div>
  )
}
```

----------------------> Todo List

```
import React,{ useState }  from 'react'
```

```
export const Todo = () =>
{
    const [tasks, setTask] = useState([]);
      const [entask, setenTask] = useState("");

      function text(e)
    {
           setenTask(e.target.value);
      }

    function addtask()
    {
        setTask([...tasks,entask])
    }

    function del(data, n)
    {
        if (n === 0)
        {
            setTask(tasks.filter((val,i) =>
            {
                if (i === n)
                {
                    return false
                }
                else
                {
                    return true
                }
            }))
        }
        if (n === 1)
        {
            setTask(tasks.filter((val,i) =>
            {
                if (val === data)
                {
                    return false
                }
                else
                {
                    return true
                }
            }))
        }

    }
      return (
           <div>
                 Enter Your Task :- <input type="text"
onChange={text}></input> &nbsp;&nbsp;
            <button onClick={addtask}> Add Task </button>
            {
                tasks.map((values, index) =>
                {
                    return <div> <br></br> <strong>Task {index+1} :-
</strong> <span>{values}</span> <button
onClick={()=>del(index,0)}>Delete By Id </button> <button
onClick={()=>del(values,1)}>Delete By Name </button> <br></br> </div>
                })
            }
           </div>
      );
}
```

-----------------------> weather.js

```
import React, {useState} from 'react'

export const Weather = () =>
{
  const [city, setCity] = useState('');
  const [weatherData, setWeatherData] = useState(null);
  const [error, setError] = useState(null);

  const cityList =
  {
    'California': { temperature: '72°F', description: 'Sunny' },
    'Texas': { temperature: '68°F', description: 'Partly Cloudy' },
    'Chicago': { temperature: '62°F', description: 'Cloudy' },
  };

  const handleCityChange = (e) =>
  {
    setCity(e.target.value);
  };

  const handleSearch = () =>
  {
    // Check if the city exists in predefined data
    if (cityList.hasOwnProperty(city))
    {
      setWeatherData(cityList[city]);
      setError(null);
    }
    else
    {
      setError(`Weather data for ${city} not found`);
      setWeatherData(null);
    }
  };

  return (
    <div>
      <h1>Weather App</h1>
      <input type="text" placeholder="Enter a city" value={city}
onChange={handleCityChange}/>
      <button onClick={handleSearch}>Search</button>

      {error && <p>{error}</p>}

      {weatherData && (
        <div>
          <h2>Weather in {city}</h2>
          <p>Temperature: {weatherData.temperature}</p>
          <p>Description: {weatherData.description}</p>
        </div>
      )}

    </div>
  );
}
```

```
----------------------------> Stock.js
```

```
import React, {useState} from 'react'

export const Stock = () =>
{
    const [name, setName] = useState('');
    const [purchasePrice, setPurchasePrice] = useState('');
    const [purchaseQuantity, setPurchaseQuantity] = useState('');
    const [sellingPrice, setSellingPrice] = useState('');
    const [sellingQuantity, setSellingQuantity] = useState('');
    const [stocks, setStocks] = useState([]);

    const handleAddStock = () =>
    {
        if (parseInt(sellingQuantity) > parseInt(purchaseQuantity))
        {
          alert('Selling quantity cannot be more than purchase
quantity.');
          return;
        }

        const stock =
        {
          name,
          purchasePrice: parseFloat(purchasePrice),
          purchaseQuantity: parseInt(purchaseQuantity),
          sellingPrice: parseFloat(sellingPrice),
          sellingQuantity: parseInt(sellingQuantity),
        };

        setStocks([...stocks, stock]);

        setName('');
        setPurchasePrice('');
        setPurchaseQuantity('');
        setSellingPrice('');
        setSellingQuantity('');
    };
  return (
    <div className="stock-detail">
      <div className="input-fields">
        <input type="text" placeholder="Name" value={name} onChange={(e)
=> setName(e.target.value)}/><br></br>
        <input type="number" placeholder="Purchase Price"
value={purchasePrice} onChange={(e) =>
setPurchasePrice(e.target.value)}/><br></br>
```

```
        <input type="number" placeholder="Purchase Quantity"
value={purchaseQuantity} onChange={(e) =>
setPurchaseQuantity(e.target.value)}/><br></br>
        <input type="number" placeholder="Selling Price"
value={sellingPrice} onChange={(e) =>
setSellingPrice(e.target.value)}/><br></br>
        <input type="number" placeholder="Selling Quantity"
value={sellingQuantity} onChange={(e) =>
setSellingQuantity(e.target.value)}/><br></br>
        <button onClick={handleAddStock}>Add Stock</button>
      </div>

      <table className="stock-table" cellPadding="10">
        <thead>
          <tr>
            <th>Name</th>
            <th>Purchase Price</th>
            <th>Purchase Quantity</th>
            <th>Selling Price</th>
            <th>Selling Quantity</th>
            <th>Profit/Loss</th>
          </tr>
        </thead>
        <tbody>
        {
          stocks.map((stock, index) => (
            <tr key={index}>
              <td>{stock.name}</td>
              <td>{stock.purchasePrice}</td>
              <td>{stock.purchaseQuantity}</td>
              <td>{stock.sellingPrice}</td>
              <td>{stock.sellingQuantity}</td>
              <td>
                  if ({stock.sellingQuantity < stock.purchaseQuantity})
                  {
                    'Invested'
                  }
                  else
                  {
                    ((stock.sellingPrice - stock.purchasePrice) *
stock.sellingQuantity).toFixed(2)
                  }
              </td>
            </tr>
          ))
        }
        </tbody>
      </table>
    </div>
  )
}
```

-------------------------> Change.js

```
import React, { useState } from 'react'

export const Change = () =>
```

```
{
    const [col, setCount] = useState("red")
    const [lj, setCount1] = useState("Hello")
    const [btn, setCount2] = useState("Hide")
    const [txt, setCount3] = useState("Good Morning")

    const changetext = () =>
    {
        if (lj === "Hello")
        {
            setCount1("Welcome")
        }
        else
        {
            setCount1("Hello")
        }
    }
    const changecolor = () =>
    {
        if (col === "red")
        {
            setCount("blue")
        }
        else
        {
            setCount("red")
        }
    }
    const Button=()=>
    {
        if (btn === "Hide")
        {
            setCount2("Show")
            setCount3("")
        }
        else
        {
            setCount2("Hide")
            setCount3("Good Morning")
        }


    }
    return (
        <div>
            <h1 style={{ color: col }}>{lj}</h1>
            <h2 style={{ color: "Black" }}>{txt}</h2>

            <button onClick={changetext}>Change text</button>
            <button onDoubleClick={changecolor}>Change Color</button>
            <button onClick={Button}>{btn}</button>
        </div>
    )

}
```

-------------------------> Clock.js

```
import React, { useEffect, useState } from 'react'

export const Clock = () =>
{
    const [time, setTime] = useState(new Date());
    useEffect(() =>
    {
        setInterval(setTime(new Date()),1000)
    });
  return (
    <div>
        <h2
style={{color:'black'}}>{time.getHours()}:{time.getMinutes()}:{time.getSe
conds()}</h2>
    </div>
  )
}
```

```
--------------------------> 22.js
```

```
const mg = require("mongoose")
const v= require("validator")

mg.connect("mongodb://127.0.0.1:27017/lju").then(()=>{console.log("Sucess
")}).catch((err)=>{console.log(err)})

const mySchema = new mg.Schema(
    {
        name:{
            type:String,
            required:true,
            lowercase:true,
            trim:true,
            minlength:[3,"Min Length 3"],
            maxlength:[10,"Max Length 10"]
        },
        email:{
            type:String,
            validate(val)
            {
                if(!v.isEmail(val))
                {
                    throw new Error("Enter Valid Email")
                }
            }
        },
        doj:{
            type: Date,
            $gte: Date("2010-01-01T00:00:00Z"),
            $lte: Date("2022-12-31T23:59:59Z")
        }
    }
)
const person = new mg.model("person",mySchema)
const createDoc = async()=>
{
    try{
        const data = new person(
        {
            name:"Sana",
            email: "sana78@gmail.com",
            doj: Date.now()
        })
        const result = await data.save();
        console.log(result)
    }
    catch{
        console.log("ERROR")
    }
}

createDoc()
```

----------------------> 23.js

```
const mg = require("mongoose")
const v= require("validator")

mg.connect("mongodb://127.0.0.1:27017/details").then(()=>{console.log("Su
cess")}).catch((err)=>{console.log(err)})

const mySchema = new mg.Schema(
    {
        name:{
            type:String,
            required:true,
            lowercase:true,
            trim:true,
            minlength:[3,"Min Length 3"],
            maxlength:[10,"Max Length 10"]
        },
        surname:
        {
            type:String
        },
        age:{
            type: Number,
        }
    }
)
const person = new mg.model("person",mySchema)

const createDoc = async()=>
{
    try{
        const data = [{name:"XYZ", surname:"PQR",age:61},
        {name:"AVN", surname:"HGJ",age:38},
        {name:"FYH", surname:"NJH",age:29},
        {name:"VGY", surname:"KHG",age:62}]
        const result = []
        result.push(await person.insertMany(data))
        result.push(await
person.updateOne({age:{$gte:60}},{$set:{category:"SeniorCitizen"}}))
        result.push(await person.find({age:{$gt:30,$lt:60}}))
        //result.push(await
person.find({},{name:0,age:0,surname:1}).sort({age:1}))
        result.push(await person.deleteMany({age:{$gt:60}}))

        console.log("Query Result",result)
    }
    catch(err)
    {
        console.log("ERROR",err)
    }
}

createDoc()
```

----------------------------------> 31.js

```
const mg = require("mongoose")
const v= require("validator")

mg.connect("mongodb://127.0.0.1:27017/mymy").then(()=>{console.log("Suces
s")}).catch((err)=>{console.log(err)})

const mySchema = new mg.Schema(
    {
        name:{
            type:String,
            required:true,
            uppercase:true,
            trim:true,
            maxlength:[12,"Max Length 10"]
        },
        email:{
            type:String,
            validate(val)
            {
                if(!v.isEmail(val))
                {
                    throw new Error("Enter Valid Email")
                }
            }
        },
        doe:{
            type: Date,
            $gte: Date("2022-10-01T00:00:00Z"),
            $lte: Date("2023-10-12T23:59:59Z")
        },
        city:
        {
            type:String,
            lowercase:true,
            enum:["ahmedabad","gandhinagar","vadodara"]
        }
    }
)
const person = new mg.model("person",mySchema)

const createDoc = async()=>
{
    try{
        const data = new person(
        {
            name:"Sana",
            email: "sana78@gmail.com",
            doe: Date.now(),
            city: "ahmedabad"
        })
        const result = await data.save();
        console.log(result)
    }
    catch{
        console.log("ERROR")
    }
}

createDoc()
```

---------------------------> 32.js

```
const mg=require("mongoose")
const v=require("validator")
mg.connect("mongodb://127.0.0.1:27017/new").then(()=>{console.log("succes
s")}).
catch((err)=>{console.error(err)});
//mg.pluralize(null)

const mySchema = new mg.Schema(
{
    _id:Number,
    name:String,
    age:Number,
    position:String,
    salary:Number
});

const emp=new mg.model("employ",mySchema)

const createDoc=async()=>
{
    try{
    const personData1=[{_id: 1,name: "Eric",age: 30,position: "Full
StackDeveloper",salary: 60000},
    {_id: 2,name: "Erica",age: 35,position: "Intern",salary: 8000},
    {_id: 3,name: "Erical",age: 40,position: "UX/UI Designer",salary:
56000},
    {_id: 4,name: "treric7",age: 37,position: "Team Leader",salary:
85000},
    {_id: 5,name: "Eliza",age: 25,position: "Software Developer",salary: 
45000},
    {_id: 6,name: "Trian",age: 29,position: "Data Scientist",salary: 
75000},
    {_id: 7,name: "Elizan",age: 25,position: "Full Stack 
Developer",salary:49000}]

    const result=[]
    result.push(await emp.insertMany(personData1))
    result.push(await emp.find())
    result.push(await emp.find({position:"Full Stack Developer"}))
    result.push(await emp.find({ age: { $gte: 30, $lte: 40 } 
},{name:1,_id:0}))
    result.push(await emp.find().sort({ salary: -1 }).limit(1))
    result.push(await emp.find({ salary: { $gt: 50000 } }))
    result.push(await emp.find({salary:{$gt:50000}}).count())
    result.push(await emp.find({ $and: [{ $or: [{ position: "Software 
Developer"}, { position: "Full Stack Developer" }] }, { age: { $lt: 30 } 
}] }))
    result.push(await emp.updateOne({salary:{$lt:50000}},{ $mul: { 
salary: 1.1 }}))
    result.push(await emp.deleteMany({ age: { $gt: 50 } }))
    result.push(await emp.updateMany({ position: "Data Scientist" }, { 
$mul: {salary: 1.05 } }))
    result.push(await emp.find({name:{$regex:/an$/}}))
    result.push(await emp.find({name:{$regex:/^eri[A-z]{2}$/i}}))

    result.push(await emp.find({name:{$regex:/ric/i}}))
    result.push(await emp.find({name:{$regex:/^[A-Za-z]{4,5}$/i}}))
    result.push(await emp.find({name:{$regex:/[0-9]$/}}))

    console.log('Query Results:', result);
    }
    catch(err) {
    console.log(err);
    }
}
createDoc()



---------------------------------------------------------->>>>>BloodDon


>>>>>>>>>>>>Blood.js


import React, {useState} from 'react'
import axios from 'axios';

export const Blood = () =>
{
    const [formData, setFormData] = useState({
        city: 'Ahmedabad', // Default value
        bloodGroup: 'O+', // Default value
      });

      const handleSubmit = async (e) => {
        e.preventDefault();
        try
        {
          // Send a POST request to your server to insert the data into
the MongoDB database
          const response = await
axios.post('http://localhost:5000/api/insertUser', formData);
          console.log('Data inserted:', response.data);
        }
        catch (error)
        {
          console.error('Error inserting data:', error);
        }
      };

      const handleChange = (e) =>
      {
        const { name, value } = e.target;
        setFormData({ ...formData, [name]: value });
      };

      return (
        <div>
          <h2>User Information Form</h2>
          <form onSubmit={handleSubmit}>
            <div>
              <label>City:</label>
              <select name="city" onChange={handleChange}>
                <option value="Ahmedabad">Ahmedabad</option>
                <option value="Rajkot">Rajkot</option>
                <option value="Surat">Surat</option>
                <option value="Vadodara">Vadodara</option>
              </select>
            </div>
            <div>
              <label>Blood Group:</label>
              <label>
                <input type="radio" name="bloodGroup" value="O+"
onChange={handleChange}/>
                O+
              </label>
              <label>
                <input type="radio" name="bloodGroup" value="A+"
onChange={handleChange}/>
                A+
              </label>
              <label>
                <input type="radio" name="bloodGroup" value="B+"
onChange={handleChange}/>
                B+
              </label>
              <label>
                <input type="radio" name="bloodGroup" value="AB+"
onChange={handleChange}/>
                AB+
              </label>
            </div>
            <button type="submit">Submit</button>
          </form>
        </div>
    );
}
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>  server.js


const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

const app = express();
const port = process.env.PORT || 5000;

// Enable CORS
app.use(cors());

// Connect to MongoDB (replace with your MongoDB connection URI)
mongoose.connect('mongodb://127.0.0.1:27017/USER',
{
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

// Define a schema for your MongoDB collection
const userSchema = new mongoose.Schema({
  city: String,
  bloodGroup: String,
});

// Create a model for the User collection
const User = mongoose.model('User', userSchema);

// Middleware to parse JSON requests
app.use(bodyParser.json());

// API route to insert user data into the MongoDB collection
app.post('/api/insertUser', async (req, res) => {
  try
  {
    const userData = req.body;
    const newUser = new User(userData);
    await newUser.save();
    res.status(201).json(newUser);
    res.end()
  }
  catch (error)
  {
    res.status(500).json({ error: 'Internal server error' });
  }
});

// Start the server
app.listen(port, () =>
{
  console.log(`Server is running on port ${port}`);
});
```

--->>>>>>>> subject form

>>>>>>>>>>>>>>>>> student.js

```
import React, { useState } from 'react';
import axios from 'axios';

const StudentForm = () =>
{
  const [formData, setFormData] = useState(
    {
    subject: 'FSD2', // Default subject
    marks: '',
  });

  const handleSubmit = async (e) =>
  {
    e.preventDefault();
    try
    {
      // Send a POST request to your server to insert the data into the
MongoDB database
      const response = await
axios.post('http://localhost:5000/api/insertStudentData', formData);
      console.log('Data inserted:', response.data);
    }
    catch (error)
    {
      console.error('Error inserting data:', error);
    }
  };

  const handleInputChange = (e) =>
  {
    const { name, value } = e.target;
    setFormData({ ...formData, [name]: value });
  };

  return (
    <div>
      <h2>Student Data Form</h2>
      <form onSubmit={handleSubmit}>
        <div>
          <label>Subject Name:</label>
          <select name="subject" onChange={handleInputChange}>
            <option value="FSD2">FSD2</option>
            <option value="FCSP2">FCSP2</option>
            <option value="DS">DS</option>
            <option value="TOC">TOC</option>
            <option value="COA">COA</option>
          </select>
        </div>
        <div>
          <label >Marks:</label>
          <input type="text" name="marks" onChange={handleInputChange}/>
        </div>
        <button type="submit">Submit</button>
      </form>
    </div>
  );
};

export default StudentForm;



>>>>>>>>>>>>>>>>>>>> server.js

const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

const app = express();
const port = process.env.PORT || 5000;

// Enable CORS
app.use(cors());

// Connect to MongoDB (replace with your MongoDB connection URI)
mongoose.connect('mongodb://127.0.0.1:27017/Marks',
{
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

// Define a schema for your MongoDB collection
const userSchema = new mongoose.Schema({
  subject: String,
  marks: Number,
});

// Create a model for the User collection
const User = mongoose.model('User', userSchema);

// Middleware to parse JSON requests
app.use(bodyParser.json());

// API route to insert user data into the MongoDB collection
app.post('/api/insertStudentData', async (req, res) => {
  try
  {
    const userData = req.body;
    const newUser = new User(userData);
    result = await newUser.save();
    console.log(result)
    res.send()
  }
  catch (error)
  {
    res.status(500).json({ error: 'Internal server error' });
  }
});

// Start the server
app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});
```

--------------------------->>>>>>>> routing

```
import React from 'react'
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import { Sub } from './Sub';
import { Page } from './Page'

export const Run = () =>
{
  return (
    <div>
      <Router>
        <ul>
          <li><Link to="/FSD">FSD</Link></li>
        </ul>
        <Routes>
          <Route path="/FSD" element={<Sub/>}></Route>
          <Route path="*" element={<Page/>}></Route>
        </Routes>
      </Router>
    </div>
  )
}
```