

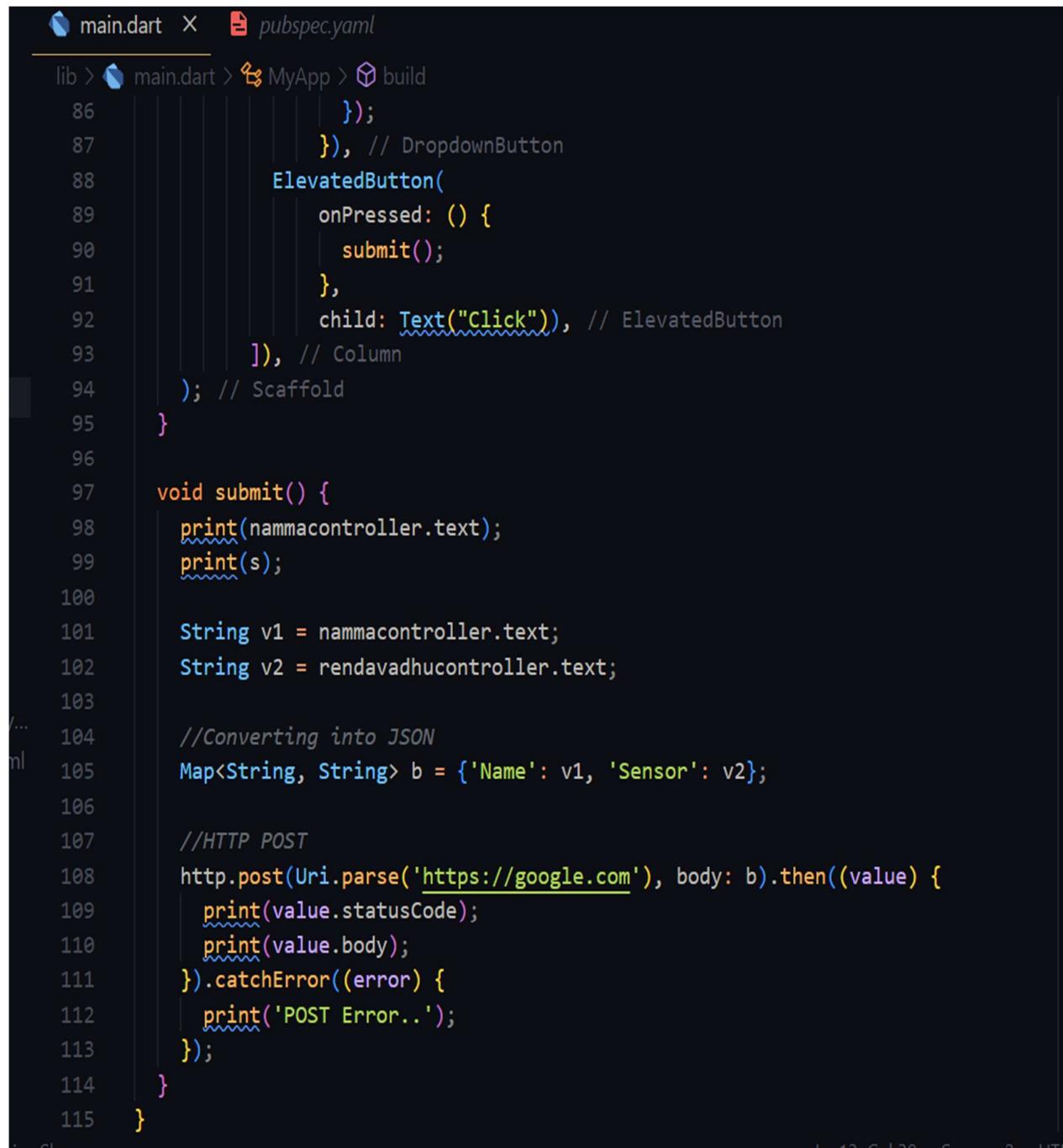
1. Flutter StatefulWidget with get and Post method:

```
main.dart  X  pubspec.yaml
lib > main.dart > MyApp > build
1 import 'package:flutter/material.dart';
2 import 'package:http/http.dart' as http;
3
4 void main() {
5   runApp(const MyApp());
6 }
7
8 class MyApp extends StatelessWidget {
9   const MyApp({super.key});
10
11   // This widget is the root of your application.
12   @override
13   Widget build(BuildContext context) {
14     return MaterialApp(
15       title: 'Flutter Demo',
16       theme: ThemeData(
17         colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
18         useMaterial3: true,
19       ), // ThemeData
20       home: Demo(),
21     ); // MaterialApp
22   }
23 }
24
25 class Demo extends StatefulWidget {
26   const Demo({super.key});
27
28   @override
29   State<Demo> createState() => _DemoState();
```

```
main.dart  X  pubspec.yaml
lib > main.dart > MyApp > build
30 }
31
32 class _DemoState extends State<Demo> {
33     bool isLoad = true;
34     String disp = "";
35
36     @override
37     void initState() {
38         http.get(Uri.parse('https://dummyjson.com/products/1')).then((value) {
39             setState(() {
40                 isLoad = false;
41                 disp = value.body;
42             });
43         });
44         super.initState();
45     }
46
47     TextEditingController nammacontroller = TextEditingController();
48     TextEditingController rendavadhucontroller = TextEditingController();
49     String? s;
50
51     @override
52     Widget build(BuildContext context) {
53         return Scaffold(
54             appBar: AppBar(
55                 title: Text('Trial'),
56             ), // AppBar
```

The screenshot shows a code editor with two tabs: `main.dart` and `pubspec.yaml`. The `main.dart` tab contains the following Dart code:

```
lib > main.dart > MyApp > build
56     body: Column(
57         mainAxisAlignment: MainAxisAlignment.center,
58         children: [
59             isLoading ? CircularProgressIndicator() : Text(display),
60             TextField(
61                 controller: nameController,
62                 decoration: InputDecoration(hintText: 'Enter your name'),
63             ), // TextField
64
65             TextField(
66                 controller: renderAdhuController,
67                 decoration: InputDecoration(hintText: 'Enter your name'),
68             ), // TextField
69
70             DropdownButton(
71                 hint: Text("Sensor Type"),
72                 items: [
73                     DropdownMenuItem(
74                         child: Text('Type1'),
75                         value: 'Type1',
76                     ), // DropdownMenuItem
77                     DropdownMenuItem(
78                         child: Text('Type2'),
79                         value: 'Type2',
80                     ) // DropdownMenuItem
81                 ],
82                 value: selectedSensorType,
83                 onChanged: (value) {
84                     setState(() {
85                         selectedSensorType = value;
```

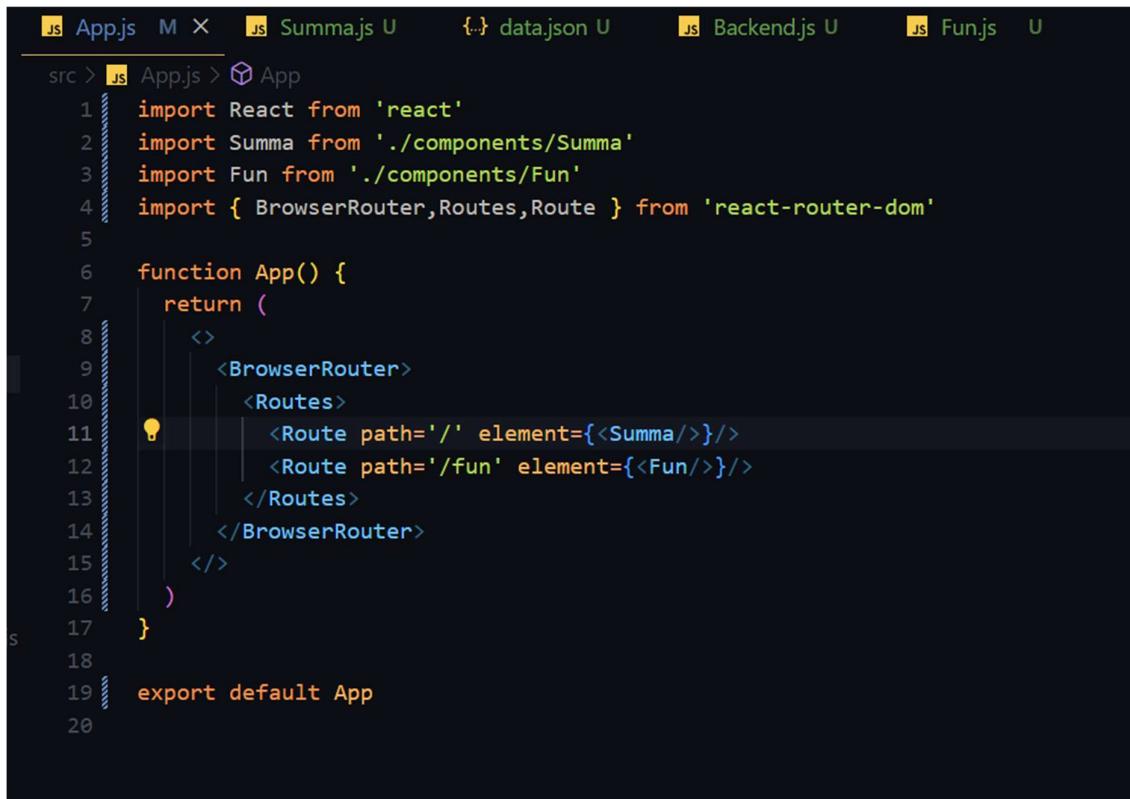


The screenshot shows a code editor with two tabs: 'main.dart' and 'pubspec.yaml'. The 'main.dart' tab is active and displays the following Dart code:

```
lib > main.dart > MyApp > build
86     );
87     }, // DropdownButton
88     ElevatedButton(
89         onPressed: () {
90             submit();
91         },
92         child: Text("Click"), // ElevatedButton
93     ), // Column
94 ); // Scaffold
95 }
96
97 void submit() {
98     print(nammacontroller.text);
99     print(s);
100
101    String v1 = nammacontroller.text;
102    String v2 = rendavadhucontroller.text;
103
104    //Converting into JSON
105    Map<String, String> b = {'Name': v1, 'Sensor': v2};
106
107    //HTTP POST
108    http.post(Uri.parse('https://google.com'), body: b).then((value) {
109        print(value.statusCode);
110        print(value.body);
111    }).catchError((error) {
112        print('POST Error..');
113    });
114 }
115 }
```

2. Sending form data from React to Node and Storing it JSON File..

React :



The screenshot shows a code editor with a dark theme. The file being edited is `App.js` located in the `src` directory. The code defines a `App` component that uses `react-router-dom` to handle routes. It imports `React`, `Summa`, `Fun`, and the necessary components from `react-router-dom`. The component returns a `BrowserRouter` with a `Routes` component containing two `Route` elements: one for the root path (`/`) pointing to `Summa`, and another for the path `/fun` pointing to `Fun`. The code ends with an `export default App` statement. Other files visible in the background include `Summa.js`, `data.json`, `Backend.js`, and `Fun.js`.

```
src > App.js > App
1 import React from 'react'
2 import Summa from './components/Summa'
3 import Fun from './components/Fun'
4 import { BrowserRouter, Routes, Route } from 'react-router-dom'
5
6 function App() {
7   return (
8     <>
9       <BrowserRouter>
10      <Routes>
11        | <Route path='/' element={<Summa/>} />
12        | <Route path='/fun' element={<Fun/>} />
13      </Routes>
14    </BrowserRouter>
15  </>
16)
17}
18
19 export default App
20
```

```
js App.js M      js Summa.js U X  {} data.json U      js Backend.js U      js Fun.js U
src > components > js Summa.js > ⚙ Summa
1 import React, { useState } from 'react'
2 import { useNavigate, Link } from 'react-router-dom'
3
4 function Summa() {
5     const nav = useNavigate();
6
7     const [lname, setlname] = useState('')
8     const [gend, setgend] = useState('')
9
10    function handlesubmit() {
11        let c = document.getElementById('fn').value
12        let jdata = {c, lname, gend}
13        fetch('http://localhost:3030/sd',
14            {
15                method: 'POST',
16                headers: {'Content-Type': 'Application/json'},
17                body: JSON.stringify(jdata, null, 3)
18            }
19        )
20    }
21}
```

```
JS App.js M JS Summa.js X {..} data.json U JS Backend.js U JS Fun.js U
src > components > JS Summa.js > Summa
23     const handleget=async()=>{
24         let result= await fetch('http://localhost:3030/sd',[method:'GET'])
25         result=await result.text()
26         document.getElementById('trial').innerHTML=result;
27     }
28
29     return (
30         <div>
31             <center>
32                 firstname:
33                 <input type='text' id='fn' placeholder='Enter your first name' />
34                 <br/>
35                 lastname:
36                 <input type='text' onBlur={(e)=>setlname(e.target.value)} placeholder='Enter your last name' />
37                 <br/>
38                 Gender:
39                 <select onChange={(e)=>setgend(e.target.value)}>
40                     <option value=''>select</option>
41                     <option value='male'>Male</option>
42                     <option value='female'>Female</option>
43                 </select>
44                 <br/>
45                 <button type='submit' onClick={handlesubmit}>Submit</button>
46             </center>
47             <button onClick={handleget}>Resp</button>
48             <div id='trial'>
49
50                 </div>
51             </div>
52         )
53     }
54     export default Summa;
```

NODE:

```
JS App.js M JS Summa.js U {} data.json U JS Backend.js U X JS Fun.js U
src > JS Backend.js > [e] server > http.createServer() callback > <function>
1  const http=require('http');
2  const cors=require('cors');
3  const fs=require('fs');
4
5  const server=http.createServer((req,res)=>{
6      cors()(req,res,()=>{
7
8          if(req.method==='POST' && req.url=='/sd'){
9              let body="";
10             req.on('data',(chunk)=>{
11                 body+=chunk;
12             })
13             fs.open('data.json','a',(err,fd)=>{
14                 if(err){
15                     res.writeHead(200)
16                     res.end('error')
17                 }
18                 else{
19                     fs.writeFileSync(fd,body)
20                     res.end('Successfully Written to File')
21                 }
22             })
23         }
24
25         else if(req.url=='/sd' && req.method=='GET'){
26             let body="";
27             body=fs.readFileSync('data.json',{encoding:'utf-8'})
28             console.log(body);
29             res.writeHead(200,['content-type':'text/plain'])
30             res.end(body);
31         }
32     })
33 })
34
35 server.listen(3030);
36
```

3. Post and Get form Data from React to Node

React:

```
js App.js M      js Summa.js U X  {} data.json U      js Backend.js U      js Fun.js U
src > components > js Summa.js > Summa
1 import React, { useState } from 'react'
2 import { useNavigate, Link } from 'react-router-dom'
3
4 function Summa() {
5     const nav = useNavigate();
6
7     const [lname, setlname] = useState('')
8     const [gend, setgend] = useState('')
9
10    function handlesubmit() {
11        let c = document.getElementById('fn').value
12        let jdata = { c, lname, gend }
13        fetch('http://localhost:3030/sd',
14            {
15                method: 'POST',
16                headers: { 'Content-Type': 'Application/json' },
17                body: JSON.stringify(jdata, null, 3)
18            }
19        )
20    }
21}
```

```
JS App.js M JS Summa.js U X -> data.json U JS Backend.js U JS Fun.js U
src > components > JS Summa.js > Summa
23     const handleget=async()=>{
24         let result= await fetch('http://localhost:3030/sd',{method:'GET'})
25         result=await result.text()
26         document.getElementById('trial').innerHTML=result;
27     }
28
29     return (
30         <div>
31             <center>
32                 firstname:
33                 <input type='text' id='fn' placeholder='Enter your first name'/>
34                 <br/>
35                 lastname:
36                 <input type='text' onBlur={(e)=>setlname(e.target.value)} placeholder='Enter your last name'/>
37                 <br/>
38                 Gender:
39                 <select onChange={(e)=>setgend(e.target.value)}>
40                     <option value=''>select</option>
41                     <option value='male'>Male</option>
42                     <option value='female'>Female</option>
43                 </select>
44                 <br/>
45                 <button type='submit' onClick={handlesubmit}>Submit</button>
46             </center>
47             <button onClick={handleget}>Resp</button>
48             <div id='trial'>
49
50                 </div>
51             </div>
52     )
52
53     }
54     export default Summa;
```

NODE:

```
...  [js] App.js M  [js] Summa.js U  ... data.json U  [js] Backend.js U X  [js] Fun.js U
src > [js] Backend.js > [o] server > [o] http.createServer() callback > [o] <function>
  1  const http=require('http');
  2  const cors=require('cors');
  3  const fs=require('fs');
  4  const { log } = require('console');
  5  let body="";
  6  const server=http.createServer((req,res)=>{
  7    cors()(req,res,()=>{
  8
  9      if(req.method==='POST' && req.url==='/sd'){
 10        req.on('data',(chunk)=>{
 11          body+=chunk;
 12        })
 13        console.log('Received..');
 14      }
 26
 27      else if(req.url==='/sd' && req.method==='GET'){
 28        res.writeHead(200,{'content-type':'text/plain'})
 29        res.end(body);
 30      }
 31    })
 32  })
 33
 34  server.listen(3030);
 35
```

4. Sending form data from React to MongoDB

Node:

```
...  [o] Backend.jsx M X
src > myComponents > [o] Backend.jsx > [o] server > [o] http.createServer() callback > [o] <function> > [o] req.on('end') callback
  1  const http = require('http');
  2  const cors = require("cors");
  3  const { MongoClient } = require('mongodb');
  4
  5  const url = "mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.0.2/StudentsDB";
  6
  7  const server = http.createServer((req, res) => {
  8    cors()(req, res, () => {
  9      let body = '';
 10
 11      req.on('data', (chunk) => {
 12        body += chunk;
 13      });
 14
 15      req.on('end', () => {
 16        try {
 17          const data = JSON.parse(body);
 18          let name = data.name;
 19          let email = data.email;
 20          let pass = data.password;
```

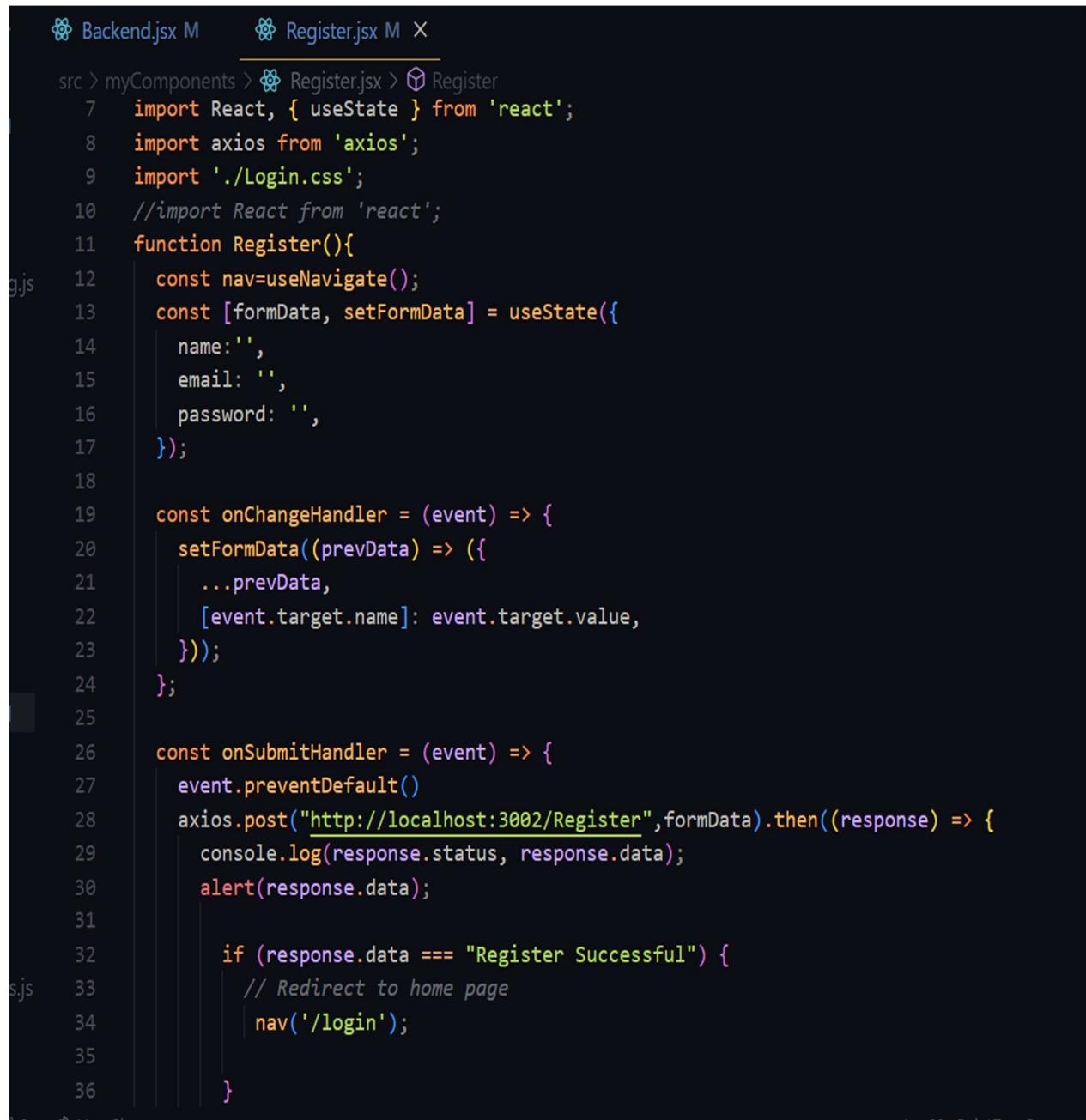
```
Backend.jsx M X
src > myComponents > Backend.jsx > [o] server > http.createServer() callback > <function> > req.on('end') callback
59  if (req.method === 'POST' && req.url === '/Register') {
60    MongoClient.connect(url)
61      .then(client => {
62        const db = client.db("StudentsDB");
63
64        db.collection("StudentsTable").insertOne({ Student_Name: name, EmailID: email, Password: pass })
65        .then(result => {
66          if (result) {
67            console.log("Register Successful");
68            res.writeHead(200, {"content-type": "text/plain"})
69            res.end("Register Successful");
70
71        } else {
72          console.log("Register Unsuccessful");
73          res.writeHead(401, {"content-type": "text/plain"})
74          res.end("Register Unsuccessful");
75
76        }
77      })
78      .catch(err => {
79        console.error("Error querying database:", err);
80        res.writeHead(500, {"content-type": "text/plain"})
81        res.end("Error querying database");
82
83      })
84      .finally(() => {
85        client.close();
86      });
87  })

```

```
Backend.jsx M X
src > myComponents > Backend.jsx > [o] server > http.createServer() callback > <function> > t
88  .catch(err => {
89    console.error("Error connecting to the database:", err);
90    res.writeHead(500, {"content-type": "text/plain"})
91    res.end("Error connecting to the database");
92
93  });
94 } else {
95   console.log("Error Post");
96   res.writeHead(404, { 'Content-Type': 'text/plain' });
97   res.end("Not Found");
98 }
99 } catch (err) {
100   console.error("Error parsing JSON:", err);
101   res.writeHead(400, { 'Content-Type': 'text/plain' });
102   res.end("Error parsing JSON");
103 }
104 });
105 );
106 );
107
108 server.listen(3002, () => {
109   console.log("Server is running on port 3002");
110 });
111

```

React:



```
Backend.jsx M Register.jsx M X
src > myComponents > Register.jsx > Register
  7 import React, { useState } from 'react';
  8 import axios from 'axios';
  9 import './Login.css';
10 //import React from 'react';
11 function Register(){
g.js
12   const nav=useNavigate();
13   const [formData, setFormData] = useState({
14     name:'',
15     email: '',
16     password: '',
17   });
18
19   const onChangeHandler = (event) => {
20     setFormData((prevData) => ({
21       ...prevData,
22       [event.target.name]: event.target.value,
23     }));
24   };
25
26   const onSubmitHandler = (event) => {
27     event.preventDefault()
28     axios.post("http://localhost:3002/Register",formData).then((response) => {
29       console.log(response.status, response.data);
30       alert(response.data);
31
32       if (response.data === "Register Successful") {
s.js
33         // Redirect to home page
34         nav('/login');
35
36     }

```

```
Backend.jsx M Registerjsx M X
src > myComponents > Registerjsx > Register
37     })
38     .catch((error) => {
39       console.error('Error:', error.message);
40       alert('Error: ' + error.message);
41     });
42   };
43   return (
44     <div>
45       <center>
46         <form onSubmit={(e)=>onSubmitHandler(e)}>
47           <div className='gbat'>
48             <PersonIcon className='g2'/>
49             <input name="name" type="text" placeholder='Enter your Name' onChange={onChangeHandler}>
50               value={formData.name}
51               required
52             />
53           </div>
54           <div className='gbat'>
55             <AccountCircleIcon className='g2'/>
56             <input name="email" placeholder='Enter your Email' onChange={onChangeHandler}>
57               value={formData.email}
58               required
59             />
60           </div>
61           <div className='gbat'>
62             <KeyIcon className='g2'/>
63             <input name='password' type='password' placeholder='Enter your Password' onChange={onChangeHandler}>
64               value={formData.password}
65               required
66             />
67           </div>
68           <button>Register</button>
69         </form>
70       </center>
71     </div>
72   )
73 }
74
75
76 export default Register
77
```

5. Flutter and Node integration (Sending Data from Mobile app and Receiving in Web App)

Flutter:

```
1 import 'package:flutter/material.dart';
2 import 'package:http/http.dart' as http;
3 Run | Debug | Profile
4 void main() {
5   runApp(const MyApp());
6 }
7 class MyApp extends StatelessWidget {
8   const MyApp({super.key});
9   // This widget is the root of your application.
10  @override
11  Widget build(BuildContext context) {
12    return MaterialApp(
13      debugShowCheckedModeBanner: false,
14      title: 'Flutter Demo',
15      theme: ThemeData(
16        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
17        useMaterial3: true,
18      ), // ThemeData
19      home: const Page(),
20    ); // MaterialApp
21  }
22 }
23 class Page extends StatefulWidget {
24   const Page({super.key});
25   @override
26   State<Page> createState() => _PageState();
27 }
28 class _PageState extends State<Page> {
```

```
lib > main.dart > _PageState > getData
27 class _PageState extends State<Page> {
28     TextEditingController textControl = TextEditingController();
29     String? subName = 'Course';
30     String data_from_GET = '';
31     bool isGETting = true;
32     @override
33     Widget build(BuildContext context) {
34         return Scaffold(
35             body: Column(
36                 mainAxisAlignment: MainAxisAlignment.center,
37                 children: [
38                     TextField(
39                         controller: textControl,
40                         decoration: const InputDecoration(hintText: 'Your Input Text'),
41                     ), // TextField
42                     DropdownButton(
43                         hint: Text('Enter Subject:'),
44                         items: const [
45                             DropdownMenuItem(value: 'SDA', child: Text('SDA')),
46                             DropdownMenuItem(value: 'IIoT', child: Text('IIoT'))
47                         ],
48                         onChanged: (value) => subName = value,
49                     ), // DropdownButton
50                     ElevatedButton(onPressed: postData, child: const Text('POST')),
51                     ElevatedButton(onPressed: getData, child: const Text('GET')),
52                     isGETting ? const CircularProgressIndicator() : Text(data_from_GET)
53                 ],
54             ), // Column
55         ); // Scaffold
```

```
56     }
57     void getData() {
58         http.get(Uri.parse('http://192.168.1.10:3030')).then((value) => {
59             setState(() {
60                 isGETting = false;
61                 data_from_GET = value.body;
62             });
63         });
64     }
65     void postData() {
66         Map<String, String?> data = {'name': textControl.text, 'course': subName};
67         http
68             .post(Uri.parse('http://192.168.1.10:3030'), body: data)
69             .then((value) => print(value.body));
70     }
71 }
72 }
```

Node:

The screenshot shows a code editor with several tabs at the top: 'Calculator.js U', 'Calculator.css U X', 'App.js M', and 'back.js U X'. The 'back.js' tab is active. The code is written in Node.js and defines a server that handles POST and GET requests. It uses the 'http' module to create a server and the 'cors' module for cross-origin requests. The POST handler concatenates data chunks into a body and logs it before sending a JSON response. The GET handler sets the content type to 'application/json' and sends the body directly. The server listens on port 3030 and logs 'Request Received' for each request.

```
src > back.js > [e] server > http.createServer() callback > <function> > req.on('end') callback
1  const http = require('http')
2  const cors = require('cors')
3
4  let body = ''
5
6  const server = http.createServer((req,res)=>{
7      cors()(req,res,()=>{
8          if(req.method==='POST'){
9              req.on('data',(chunk)=>body+=chunk)
10             req.on('end',()=>{
11                 console.log(body);
12                 res.end(JSON.stringify({message:'Data Received'}))
13             })
14         }
15         else if(req.method==='GET'){
16             res.writeHead(200,{'Content-Type':'application/json'})
17             res.end(body)
18         }
19     })
20 })
21
22 server.listen(3030,()=>console.log('Server running on port 3030'))
23 server.on('request',()=>console.log('Request Received'))
```

6. Calculator Program with React

Calculator.js

```
src > components > Calculator.js > [F8] Fun
  1 import React, { useState } from 'react';
  2 import './Calculator.css';
  3
  4 const Fun = () => {
  5   const [input, setInput] = useState('');
  6
  7   const handleButtonClick = (value) => {
  8     setInput(input + value);
  9   };
 10
 11   const handleCalculate = () => {
 12     try {
 13       setInput(eval(input).toString());
 14     } catch (error) {
 15       setInput('Error');
 16     }
 17   };
 18
 19   const handleClear = () => {
 20     setInput('');
 21   };
 22
 23   return (
 24     <div className="calculator">
 25       <div className="display">{input}</div>
 26       <div className="buttons">
 27         <button onClick={() => handleButtonClick('1')}>1</button>
 28         <button onClick={() => handleButtonClick('2')}>2</button>
 29         <button onClick={() => handleButtonClick('3')}>3</button>
 30         <button onClick={() => handleButtonClick('+')}>+</button>
```

```
JS Calculator.js U X CSS Calculator.css U X
src > components > JS Calculator.js > [o] Fun
  31   <button onClick={() => handleButtonClick('4')}>4</button>
  32   <button onClick={() => handleButtonClick('5')}>5</button>
  33   <button onClick={() => handleButtonClick('6')}>6</button>
  34   <button onClick={() => handleButtonClick('-')}>-</button>
  35   <button onClick={() => handleButtonClick('7')}>7</button>
  36   <button onClick={() => handleButtonClick('8')}>8</button>
  37   <button onClick={() => handleButtonClick('9')}>9</button>
  38   <button onClick={() => handleButtonClick('*')}>*</button>
  39   <button onClick={() => handleButtonClick('0')}>0</button>
  40   <button onClick={() => handleClear()}>C</button>
  41   <button onClick={() => handleCalculate()}>=</button>
  42   <button onClick={() => handleButtonClick('/')}>/</button>
  43 </div>
  44 </div>
  45 );
  46 };
  47
  48 function Calculator() {
  49   return (
  50     <div className="App">
  51       <header className="App-header">
  52         <Fun />
  53       </header>
  54     </div>
  55   );
  56 }
  57
  58 export default Calculator;
```

Calculator.css

```
JS Calculator.js U      CSS Calculator.css U X
src > components > Calculator.css > button:hover
1   .calculator {
2     width: 300px;
3     margin: 50px auto;
4     border: 1px solid #ccc;
5     border-radius: 5px;
6     padding: 10px;
7     box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
8   }
9
10  .display {
11    font-size: 20px;
12    margin-bottom: 10px;
13    padding: 10px;
14    border: 1px solid #ccc;
15    border-radius: 5px;
16    text-align: right;
17    height: 28px;
18  }
19
20  .buttons {
21    display: grid;
22    grid-template-columns: repeat(4, 1fr);
23    gap: 10px;
24  }
25
26  button {
27    font-size: 18px;
28    padding: 10px;
29    border: 1px solid #ccc;
30    border-radius: 5px;
31    cursor: pointer;
32    background-color: #f8f8f8;
33  }
34
35  button:hover {
36    background-color: #e0e0e0;
37  }
```